

# MAE 5905: Introdução à Ciência de Dados

Pedro A. Morettin

Instituto de Matemática e Estatística  
Universidade de São Paulo  
pam@ime.usp.br  
<http://www.ime.usp.br/~pam>

## Aula 1

13 de março de 2023

# Sumário

1 As Origens

2 Inferência Bayesiana

3 Inferência Frequentista

4 Era Moderna

5 Estatística

# Paradigma

1. Modelo, padrão a ser seguido.
2. Um pressuposto filosófico, uma teoria, um conhecimento, que origina o estudo de um campo científico.
3. Aquilo que os membros de uma comunidade científica partilham.

Exemplos:

- Inferência Frequentista, Inferência Bayesiana
- Data Mining, Neural Networks, Data Science
- Statistical Learning, Machine Learning

# Aprendizado com Estatística

- **Aprendizado com Estatística (AE)/Statistical Learning (SL)**: nomenclatura nova, mas a maioria dos conceitos foram desenvolvidos desde o Século 19. Métodos estatísticos para previsão, classificação, análise de agrupamentos etc. Inferência é o objetivo e interpretação é importante.
- **Aprendizado com (ou de) Máquina (AM)/Machine Learning (ML)**: métodos para “aprender” padrões ocultos em dados. Usados para previsão, classificação, reconhecimento de padrões, análise de agrupamentos etc. Pouca atenção à inferência (do ponto de vista computacional) e à interpretabilidade.
- When ML methods are statistically sound they are called **Statistical Learning (SL)** methods.
- Nosso foco: Métodos de AE.

# Probabilidade

1. Início em 1654 com Fermat (1601-1665), Pascal (1623-1662): jogos de dados
2. Huygens (1629-1695): primeiro livro de probabilidade em 1657.
3. Bayes (1702-1761): primeira versão do Teorema de Bayes, publicado em 1763.

## Gauss e Legendre

1. Gauss (1777-1856) inventou o método de mínimos quadrados (MQ) na última década do século 18 (1795) e o usou regularmente depois de 1801 em cálculos astronômicos.
2. Legendre (1752-1833): publicou no apêndice de "Nouvelles Methodes pour la Détermination des Orbites des Comètes". Nenhuma justificação.
3. Gauss (1809): deu justificativa probabilística do método. Em "The Theory of the Motion of Heavenly Bodies".
4. Implementaram o que é hoje chamado de **regressão linear**.

## Bayes ou Laplace?

1. Laplace (1749-1761): desenvolveu o Teorema de Bayes independentemente, publicado em 1774.
2. 1812: Théorie Analytique des Probabilités: aplicações científicas e práticas.
3. 1814: Essais Philosophiques sur les Probabilités: interpretação Bayesiana das probabilidades
4. Inferência Bayesiana ↔ Inferência Laplaciana. Usada a partir de 1800.
5. Fisher e Neyman: início do século 20.
6. Jeffrey (1939). Theory of Probability. Considerado como o re-início da Inferência Bayesiana
7. de Finetti, Savage, Lindley etc.

## Fisher e Neyman

1. Inferência Frequentista (testes de hipóteses, estimativa, planejamento de experimentos e amostragem) foi iniciada por R. Fisher( 1890-1962) e J. Neyman (1894-1981).
2. Fisher (1925): Statistical Methods for Research Workers. (14<sup>a</sup> Edição: 1970)
3. Fisher (1935): The Design of Experiments (8<sup>a</sup> Edição: 1966)
4. Fisher (1936): propõe a **análise discriminante linear**.

## Gosset/Student

1. W. Gosset (1876-1937): Em 1908 publicou sob o pseudônimo de Student um artigo que iniciou um novo paradigma em "Pequenas Amostras". Resultado provado por Fisher em 1912-1915.
2. Student (1908). The probable error of a mean. *Biometrika*, 6, 1-25.
3. Fisher (1922). On the mathematical foundation of theoretical statistics. *Phil. Trans. Royal Society*, A, 222, 308-368.
4. Stigler: "The most influencial article on Theoretical Statistics in the 20th Century"
5. Hald: "For the first time in the history of Statistics a framework for frequency-based general theory of parametric statistical inference was clearly formulated."

## Neyman e Pearson

1. Neyman and Pearson (1933a) On the problem of the most efficient tests of statistical hypotheses. *Philos. Trans. Royal Society, A*, 231, 289-331.
2. Neyman and Pearson (1933b). On the testing of statistical hypothesis in relation to probabilities a priori. *Proc. Cambridge Philos. Society*, 24, 429–510.
3. Livros de E. Lehmann sobre estimação e testes de hipóteses. 1967, 1983.
4. Era do "Small Data".

1940 - 2000

1. 1940: propostas de abordagens alternativas: regressão logística.
2. 1970: Nelder e Wedderburn: generalized linear models para uma classe de métodos que incluem regressão linear e logística como casos especiais.
3. 1970: Efrom: bootstrap; Hoerl e Kennard: ridge regression.
4. até o final de 1970: métodos lineares.
5. 1980: tecnologia computacional possibilita a aplicação de métodos não lineares.
6. 1984: Friedman et al. introduzem CART (Classification and regression trees) e propõem uma implementação prática de método para seleção de modelos, incluindo CV (cross validation)
7. 1986: Hastie e Tibshirani: estendem os MLG pra os modelos GAM (generalized additive models).
8. 1996: Tibshirani: introduz o LASSO; extensões para outros métodos de regularização.

## O que é Estatística?

- [1] Coleta de dados: amostras, planejamento de experimentos, estudos observacionais.
- [2] Modelagem e análise de dados.
- [3] Tomada de decisões.

## Eras da Estatística

A história da Estatística pode ser dividida em três eras:

- (1) A era de Quetelet (astrônomo, matemático, estatístico belga, 1796-1874) e seus sucessores, na qual o objetivo era obter grandes conjuntos de dados (censos) em ciências sociais.
- (2) O período clássico de Pearson (1857-1936), Fisher (1890-1962), Neyman (1894-1981), Hotelling (1895-1973) a seus sucessores, que desenvolveram a teoria de inferência ótima; métodos apropriados para *small data sets*.
- (3) A era da produção de dados em massa (*big data sets*), com novas tecnologias como *microarrays*, dados de alta frequência em finanças, dados astronômicos etc.

# Algoritmo e Inferência

## Análise Estatística:

- (a) algoritmica
- (b) inferencial

**Exemplo:** Considere estimar a média  $\mu = E(X)$  de uma v.a.  $X$ , definida sobre uma população.

Para uma AAS  $X_1, \dots, X_n$ , considere o estimador

$$\bar{X} = \frac{1}{n} \sum_{i=1}^n X_i.$$

Este é o **algoritmo**.

Quão acurado e preciso é  $\bar{X}$ . Esta é a parte da **inferência**.

# Algoritmo e Inferência

Algoritmos: é o que os estatísticos fazem.

Inferência: porque os estatísticos usam os algoritmos. (Efrom, 2016).

Conjuntos de dados enormes (*Big Data*) requerem novas metodologias. Esta demanda está sendo atendida por algoritmos estatísticos baseados em computação intensiva.

## Estatística e Computação

- Avanços em Estatística diretamente relacionados com avanços na área computacional.
  - → 1960: máquinas de calcular manuais, elétricas, eletrônicas.
  - 1960→ 1980: "grandes computadores": IBM 1620, CDC 360, VAX etc; cartões e discos magnéticos; FORTRAN.
  - 1980→: computadores pessoais; supercomputadores; computação paralela; "clouds"; C, C+, S.
  - Pacotes estatísticos: S-Plus, SPSS, Minitab etc. Repositório R.
  - Era do "Big Data" e da "Data Science".

## Estatística e Computação

- Avanços em Estatística diretamente relacionados com avanços na área computacional.
- → 1960: máquinas de calcular manuais, elétricas, eletrônicas.
- 1960→ 1980: "grandes computadores": IBM 1620, CDC 360, VAX etc; cartões e discos magnéticos; FORTRAN.
- 1980→: computadores pessoais; supercomputadores; computação paralela; "clouds"; C, C+, S.
- Pacotes estatísticos: S-Plus, SPSS, Minitab etc. Repositório R.
- Era do "Big Data" e da "Data Science".

## Estatística e Computação

- Avanços em Estatística diretamente relacionados com avanços na área computacional.
- → 1960: máquinas de calcular manuais, elétricas, eletrônicas.
- 1960→ 1980: "grandes computadores": IBM 1620, CDC 360, VAX etc; cartões e discos magnéticos; FORTRAN.
- 1980→: computadores pessoais; supercomputadores; computação paralela; "clouds"; C, C<sub>+</sub>, S.
- Pacotes estatísticos: S-Plus, SPSS, Minitab etc. Repositório R.
- Era do "Big Data" e da "Data Science".

## Estatística e Computação

- Avanços em Estatística diretamente relacionados com avanços na área computacional.
- → 1960: máquinas de calcular manuais, elétricas, eletrônicas.
- 1960→ 1980: "grandes computadores": IBM 1620, CDC 360, VAX etc; cartões e discos magnéticos; FORTRAN.
- 1980→: computadores pessoais; supercomputadores; computação paralela; "clouds"; C, C<sub>+</sub>, S.
- Pacotes estatísticos: S-Plus, SPSS, Minitab etc. Repositório R.
- Era do "Big Data" e da "Data Science".

## Estatística e Computação

- Avanços em Estatística diretamente relacionados com avanços na área computacional.
- → 1960: máquinas de calcular manuais, elétricas, eletrônicas.
- 1960→ 1980: "grandes computadores": IBM 1620, CDC 360, VAX etc; cartões e discos magnéticos; FORTRAN.
- 1980→: computadores pessoais; supercomputadores; computação paralela; "clouds"; C, C<sub>+</sub>, S.
- Pacotes estatísticos: S-Plus, SPSS, Minitab etc. Repositório R.
- Era do "Big Data" e da "Data Science".

## Estatística e Computação

- Avanços em Estatística diretamente relacionados com avanços na área computacional.
- → 1960: máquinas de calcular manuais, elétricas, eletrônicas.
- 1960→ 1980: "grandes computadores": IBM 1620, CDC 360, VAX etc; cartões e discos magnéticos; FORTRAN.
- 1980→: computadores pessoais; supercomputadores; computação paralela; "clouds"; C, C<sub>+</sub>, S.
- Pacotes estatísticos: S-Plus, SPSS, Minitab etc. Repositório R.
- Era do "Big Data" e da "Data Science".

## Referências

Bühlmann, P. and van de Geer, S. (2011). *Statistics for High-Dimensional Data*. Berlin: Springer.

Efron, B. and Hastie, T. (2016). *Computer Age Statistical Inference*. Cambridge University Press.

Hastie, T., Tibshirani, R. and Friedman, J. (2017). *The Elements of Statistical Learning*. Second Edition. Springer

James, G., Witten, D., Hastie, T. and Tibshirani, R. (2017). *Introduction to Statistical Learning*. Springer.

Morettin, P.A. and Singer, J.M. (202). *Estatística e Ciência de Dados*. Rio de Janeiro:LTC.

# MAE 5905: Introdução à Ciência de Dados

Pedro A. Morettin

Instituto de Matemática e Estatística  
Universidade de São Paulo  
[pam@ime.usp.br](mailto:pam@ime.usp.br)  
<http://www.ime.usp.br/~pam>

## Aula 2

23 de março de 2023

# Sumário

## 1 Estatística e Ciência de Dados

## 2 Aprendizado com Estatística e com Máquina

# Ciência de Dados

- Atualmente, os termos *Data Science* (Ciência de Dados) e *Big Data* (Megadados) são utilizados em profusão, como se fossem conceitos novos, distintos daqueles com que os estatísticos lidam há cerca de dois séculos.
- Na década de 1980, numa palestra na Universidade de Michigan, EUA, C.F. Jeff Wu já sugeria que se adotassem os rótulos *Statistical Data Science*, ou simplesmente, *Data Science*, em lugar de *Statistics*, para dar maior visibilidade ao trabalho dos estatísticos.
- Talvez seja Tukey (1962, 1977), sob a denominação *Exploratory Data Analysis* (Análise Exploratória de Dados), o primeiro a dar importância ao que hoje se chama Ciência de Dados, sugerindo que se desse mais ênfase ao uso de tabelas, gráficos e outros dispositivos para uma análise preliminar de dados, antes que se passasse a uma **análise confirmatória**, que seria a **inferência estatística**.

# Ciência de Dados

- Outros autores, como Chambers (1993), Breiman (2001) e Cleveland (1985, 1993, 2001), também enfatizaram a preparação, apresentação e descrição dos dados como atividades preparatórias para inferência ou modelagem.
- Basta uma procura simples na Internet para identificar novos centros de Ciências de Dados (CD) em várias universidades ao redor do mundo, com programas de mestrado, doutorado e mesmo graduação.
- O interessante é que muitos desses programas estão alojados em escolas de Engenharia, Bioestatística, Ciência da Computação, Administração, Economia etc., e não em departamentos de Estatística.
- Paradoxalmente, há estatísticos que acham que Estatística é a parte menos importante de CD! Certamente isso é um equívoco. Como ressalta Donoho (2017), se uma das principais características de CD é analisar grandes conjuntos de dados (Megadados), há mais de 200 anos os estatísticos têm se preocupado com a análise de vastos conjuntos de dados provenientes de censos, coleta de informações meteorológicas, observação de séries de índices financeiros etc., que têm essa característica.

# Ciência de Dados

- Outro equívoco consiste em imaginar que a Estatística Clássica (frequentista, bayesiana etc.) trata somente de pequenos volumes de dados, conhecidos como *Small Data*.
- Essa interpretação errônea vem do fato de que muitos livros didáticos apresentam conjuntos de dados, em geral de pequeno ou médio porte, para que as metodologias apresentadas possam ser aplicadas pelos leitores, mesmo utilizando calculadoras ou aplicativos estatísticos (pacotes). Nada impede que essas metodologias sejam aplicadas a grandes volumes de dados a não ser pelas dificuldades computacionais inerentes.
- Talvez seja este aspecto computacional, aquele que mascara os demais componentes daquilo que se entende por CD, pois em muitos casos, o interesse é dirigido apenas para o desenvolvimento de algoritmos cuja finalidade é aprender a partir dos dados, omitindo-se características estatísticas.

# Ciência de Dados

- Ciência de Dados(CD) é "filha" da Estatística e da Ciência da Computação.
- Perspectiva não é nova: Tukey (1962): The future of Data Analysis, AMS.
- Cientistas de diversas disciplinas estão sendo confrontados com conjuntos enormes de dados: sequenciamento genético, grandes arquivos de textos, dados astronômicos, dados financeiros de alta frequência.
- Perspectiva da Estatística, da Computação e Humana.
- Ciência de Dados: redes neurais, support vector machines, machine learning, deep learning, classification and regression trees (CART), random forests etc.

# Ciência de Dados

- Ciência de Dados(CD) é "filha" da Estatística e da Ciência da Computação.
- Perspectiva não é nova: Tukey (1962): The future of Data Analysis, AMS.
- Cientistas de diversas disciplinas estão sendo confrontados com conjuntos enormes de dados: sequenciamento genético, grandes arquivos de textos, dados astronômicos, dados financeiros de alta frequência.
- Perspectiva da Estatística, da Computação e Humana.
- Ciência de Dados: redes neurais, support vector machines, machine learning, deep learning, classification and regression trees (CART), random forests etc.

# Ciência de Dados

- Ciência de Dados(CD) é "filha" da Estatística e da Ciência da Computação.
- Perspectiva não é nova: Tukey (1962): The future of Data Analysis, AMS.
- Cientistas de diversas disciplinas estão sendo confrontados com conjuntos enormes de dados: sequenciamento genético, grandes arquivos de textos, dados astronômicos, dados financeiros de alta frequência.
- Perspectiva da Estatística, da Computação e Humana.
- Ciência de Dados: redes neurais, support vector machines, machine learning, deep learning, classification and regression trees (CART), random forests etc.

# Ciência de Dados

- Ciência de Dados(CD) é "filha" da Estatística e da Ciência da Computação.
- Perspectiva não é nova: Tukey (1962): The future of Data Analysis, AMS.
- Cientistas de diversas disciplinas estão sendo confrontados com conjuntos enormes de dados: sequenciamento genético, grandes arquivos de textos, dados astronômicos, dados financeiros de alta frequência.
- Perspectiva da Estatística, da Computação e Humana.
- Ciência de Dados: redes neurais, support vector machines, machine learning, deep learning, classification and regression trees (CART), random forests etc.

# Ciência de Dados

- Ciência de Dados(CD) é "filha" da Estatística e da Ciência da Computação.
- Perspectiva não é nova: Tukey (1962): The future of Data Analysis, AMS.
- Cientistas de diversas disciplinas estão sendo confrontados com conjuntos enormes de dados: sequenciamento genético, grandes arquivos de textos, dados astronômicos, dados financeiros de alta frequência.
- Perspectiva da Estatística, da Computação e Humana.
- Ciência de Dados: redes neurais, support vector machines, machine learning, deep learning, classification and regression trees (CART), random forests etc.

## CD: Perspectiva da Estatística

- Estatística "serve" a Ciência guiando na coleta e análise de dados.
- Dados envolvem incertezas: como foram coletados, medidos ou como foram gerados. A modelagem estatística ajuda a quantificar e racionalizar incertezas de maneira sistemática.
- Conjuntos de dados são complexos: tipos diferentes de dependência (ao longo do tempo, sobre escalas espaciais, entre variáveis diferentes)
- Dados de alta dimensão: medimos milhares de variáveis para cada unidade amostral.

## CD: Perspectiva da Estatística

- Estatística "serve" a Ciência guiando na coleta e análise de dados.
- Dados envolvem incertezas: como foram coletados, medidos ou como foram gerados. A modelagem estatística ajuda a quantificar e racionalizar incertezas de maneira sistemática.
- Conjuntos de dados são complexos: tipos diferentes de dependência (ao longo do tempo, sobre escalas espaciais, entre variáveis diferentes)
- Dados de alta dimensão: medimos milhares de variáveis para cada unidade amostral.

## CD: Perspectiva da Estatística

- Estatística "serve" a Ciência guiando na coleta e análise de dados.
- Dados envolvem incertezas: como foram coletados, medidos ou como foram gerados. A modelagem estatística ajuda a quantificar e racionalizar incertezas de maneira sistemática.
- Conjuntos de dados são complexos: tipos diferentes de dependência (ao longo do tempo, sobre escalas espaciais, entre variáveis diferentes)
- Dados de alta dimensão: medimos milhares de variáveis para cada unidade amostral.

## CD: Perspectiva da Estatística

- Estatística "serve" a Ciência guiando na coleta e análise de dados.
- Dados envolvem incertezas: como foram coletados, medidos ou como foram gerados. A modelagem estatística ajuda a quantificar e racionalizar incertezas de maneira sistemática.
- Conjuntos de dados são complexos: tipos diferentes de dependência (ao longo do tempo, sobre escalas espaciais, entre variáveis diferentes)
- Dados de alta dimensão: medimos milhares de variáveis para cada unidade amostral.

## CD: Perspectiva da Computação

- Particularmente importante na análise de dados contemporâneos, onde frequentemente nos deparamos com a dicotomia entre acurácia e precisão estatística e recursos computacionais (tempo e memória).
- Exemplos: otimização, bootstrap, MCMC.
- Distribuição de conjuntos de dados enormes por múltiplos processadores (velocidade) e múltiplos equipamentos de armazenamento (memória).

## CD: Perspectiva da Computação

- Particularmente importante na análise de dados contemporâneos, onde frequentemente nos deparamos com a dicotomia entre acurácia e precisão estatística e recursos computacionais (tempo e memória).
- Exemplos: otimização, bootstrap, MCMC.
- Distribuição de conjuntos de dados enormes por múltiplos processadores (velocidade) e múltiplos equipamentos de armazenamento (memória).

## CD: Perspectiva da Computação

- Particularmente importante na análise de dados contemporâneos, onde frequentemente nos deparamos com a dicotomia entre acurácia e precisão estatística e recursos computacionais (tempo e memória).
- Exemplos: otimização, bootstrap, MCMC.
- Distribuição de conjuntos de dados enormes por múltiplos processadores (velocidade) e múltiplos equipamentos de armazenamento (memória).

## CD: Perspectiva Humana

- CD liga modelos estatísticos e métodos computacionais para resolver problemas específicos de outras disciplinas.
- Entender o domínio de um problema, decidir quais dados obter, como processá-los, explorar e visualizar os dados, selecionar um modelo estatístico e métodos computacionais apropriados, comunicar os resultados da análise.
- Estas habilidades não são usualmente ensinadas em disciplinas tradicionais de Estatística ou Computação, mas são adquiridas por meio da experiência e colaboração com outros pesquisadores.

## CD: Perspectiva Humana

- CD liga modelos estatísticos e métodos computacionais para resolver problemas específicos de outras disciplinas.
- Entender o domínio de um problema, decidir quais dados obter, como processá-los, explorar e visualizar os dados, selecionar um modelo estatístico e métodos computacionais apropriados, comunicar os resultados da análise.
- Estas habilidades não são usualmente ensinadas em disciplinas tradicionais de Estatística ou Computação, mas são adquiridas por meio da experiência e colaboração com outros pesquisadores.

## CD: Perspectiva Humana

- CD liga modelos estatísticos e métodos computacionais para resolver problemas específicos de outras disciplinas.
- Entender o domínio de um problema, decidir quais dados obter, como processá-los, explorar e visualizar os dados, selecionar um modelo estatístico e métodos computacionais apropriados, comunicar os resultados da análise.
- Estas habilidades não são usualmente ensinadas em disciplinas tradicionais de Estatística ou Computação, mas são adquiridas por meio da experiência e colaboração com outros pesquisadores.

# Aprendizado com Estatística

- O Aprendizado com Estatística (AE) pode ser **supervisionado** ou não **supervisionado**.
- No AE supervisionado, o objetivo é prever o valor de uma variável resposta (*output*) a partir de variáveis preditoras (*inputs*).
- A variável resposta pode ser quantitativa ou qualitativa. No caso de variáveis respostas quantitativas, um dos modelos estatísticos mais utilizados é o de **regressão**; quando a variável resposta é qualitativa, utilizam-se geralmente modelos de **regressão logística** para a análise.
- Adicionalmente, para variáveis qualitativas (categóricas), com valores em um conjunto finito, os modelos mais comuns são os de classificação, em que a partir de um conjunto  $(x_i, y_i), i = 1 \dots, N$  de dados, chamado de **conjunto de treinamento**, obtemos, por exemplo, obtemos uma regra de classificação.

## Aprendizado com Estatística

- No caso de AE não supervisionado, temos apenas um conjunto de variáveis (*inputs*) e o objetivo é descrever associações e padrões entre essas variáveis. Nesse caso, não há uma variável resposta.
- Um algoritmo de AE não supervisionado pode ter por objetivo aprender a distribuição de probabilidades que gerou os dados, para efeito de estimativa de densidades, por exemplo.
- Dentre as técnicas mais utilizadas nesta situação temos a **análise de agrupamentos**, a **análise de componentes principais** e a **análise de componentes independentes** (ambas proporcionando a redução da dimensionalidade dos dados).

# Inteligência Artificial

- Inteligência Artificial (IA) é um tópico de extremo interesse e que aparece frequentemente nas mídias escritas e faladas. Normalmente o termo suscita questões do tipo: computadores no futuro vão se tornar inteligentes e a raça humana será substituída por eles? Ou que todos perderemos nossos empregos, por que seremos substituídos por robôs inteligentes? Pelo menos até o presente esses receios são infundados.
- Acredita-se que o artigo de Turing (1950) seja o primeiro a tratar do tema. A primeira frase do artigo diz:

I propose to consider the question, “Can machines think?

- De modo informal, a IA é um esforço para automatizar tarefas intelectuais usualmente realizadas por seres humanos.
- Jordan (2019). Segundo esse autor, o que é rotulado hoje como IA, nada mais é do que aquilo que chamamos de Aprendizado de Máquina (ML).

# Inteligência Artificial

Jordan (2019): Harvard Data Science Review, Issue 1.

- The problem had to do not just with data analysis, but with what database researchers call provenance—broadly, where did data arise, what inferences were drawn from the data, and how relevant are those inferences to the present situation?
- I'm also a computer scientist, and it occurred to me that the principles needed to build planetary-scale inference-and-decision-making systems of this kind, blending computer science with statistics, and considering human utilities, were nowhere to be found in my education.
- It occurred to me that the development of such principle—which will be needed not only in the medical domain but also in domains such as commerce, transportation, and education—were at least as important as those of building AI systems that can dazzle us with their game-playing or sensorimotor skills.
- This new engineering discipline will build on ideas that the preceding century gave substance to, such as **information**, **algorithm**, **data**, **uncertainty**, **computing**, **inference**, and **optimization**. Moreover, since much of the focus of the new discipline will be on data from and about humans, **its development will require perspectives from the social sciences and humanities**.

# Inteligência Artificial

- While the building blocks are in place, the principles for putting these blocks together are not, and so the blocks are currently being put together in ad-hoc ways.
- Humans are proceeding with the building of societal-scale, inference-and-decision-making systems that involve machines, humans, and the environment.
- Just as early buildings and bridges sometimes fell to the ground—in unforeseen ways and with tragic consequences—many of our early societal-scale inference-and-decision-making systems are already exposing serious conceptual flaws.
- Unfortunately, we are not very good at anticipating what the next emerging serious flaw will be. What we're missing is an engineering discipline with principles of analysis and design.

# Inteligência Artificial

- Most of what is labeled AI today, particularly in the public sphere, is actually machine learning (ML), a term in use for the past several decades.
- ML is an algorithmic field that blends ideas from statistics, computer science and many other disciplines to design algorithms that process data, make predictions, and help make decisions.
- The phrase data science emerged to refer to this phenomenon, reflecting both the need of ML algorithms experts to partner with database and distributed-systems experts to build scalable, robust ML systems, as well as reflecting the larger social and environmental scope of the resulting systems.
- This confluence of ideas and technology trends has been rebranded as AI over the past few years.

# Inteligência Artificial

- Three types of AI:
  - (i) **Human-imitative AI** : the artificially-intelligent entity should be one of us, if not physically then at least mentally;
  - (ii) **Intelligence Augmentation (IA)**: computation and data are used to create services that augment human intelligence and creativity, eg, natural language translation, which augments the ability of a human to communicate;
  - **Intelligent Infrastructure (II)**: a web of computation, data and physical entities exists that makes human environments more supportive, interesting and safe.
- We are very far from realizing human-imitative AI aspirations, that gives rise to levels of over-exuberance and media attention that is not present in other areas of engineering.
- Success in these domains is neither sufficient nor necessary to solve important IA and II problems.

# Aprendizado de Máquina=ML

- A IA está intimamente ligada ao desenvolvimento da computação (ou programação de computadores) e até a década de 1980, a IA era entendida como na **programação clássica**: temos um sistema computacional (SC) (um computador ou um *cluster* de computadores ou nuvem etc.) no qual se alimentam dados e uma regra de cálculo e se obtém uma resposta.
- Exemplo: regressão, usando-se MQ para se obter os EMQ. A regra de cálculo é um algoritmo que resolve o problema e pode ser programado em alguma linguagem (Fortran, C, S etc). A maioria dos pacotes computacionais existentes funciona dessa maneira.
- A partir da década de 1990, o aprendizado de máquina (AM-ML) criou um novo paradigma. A programação clássica não resolve problemas mais complicados, como reconhecimento de imagens, voz, escrita etc.

# Aprendizado de Máquiana=ML

- Então a ideia é **treinar** um SC no lugar de programá-lo. Isso significa que se apresentam muitos exemplos relevantes a determinada tarefa (**dados de treinamento**) ao SC, de modo que esse encontre uma estrutura estatística nesses exemplos, produzindo uma regra automatizada. Ou seja, no AM, a entrada é constituída de dados e respostas, e a saída é uma regra de cálculo. Com um novo conjunto de observações (**dados de teste**) procura-se obter a eficácia do método segundo algum critério.
- Existem atualmente, muitos procedimentos que são usados em AM (ou em AE): SVM (*support vector machines*), métodos baseados em árvores de decisão (árvores, florestas, *bagging*, *boosting*), redes neurais etc. O objetivo é obter algoritmos que tenham um alto valor preditivo em problemas de regressão, agrupamento, classificação e previsão.
- AM está fortemente relacionado com Estatística Computacional, que também trata de fazer previsões com o auxílio de computador. Tem relação forte com otimização, que fornece métodos, teoria e aplicações a este campo.

# Redes Neuronais

- As contribuições pioneiras para a área de Redes Neuronais (RN) (também denominadas redes neurais) foram as de McCulloch e Pitts (1943), que introduziram a ideia de RN como máquinas computacionais, de Hebb (1949), por postular a primeira regra para aprendizado organizado e Rosenblatt (1958), que introduziu o *perceptron*, como o primeiro modelo de aprendizado supervisionado.
- O **algoritmo do perceptron** (programado para o IBM 704) foi implementado por uma máquina, chamada Mark I, planejada para reconhecimento de imagens. O modelo consiste de uma combinação linear das entradas, incorporando um viés externo. A soma resultante é aplicada a um limitador, na forma de uma função degrau (ou uma sigmóide).
- Se  $\mathbf{x} = (+1, x_1, x_2, \dots, x_p)^\top$  contém as entradas,  $\mathbf{w} = (b, w_1, w_2, \dots, w_p)^\top$  são os pesos, a saída é dada por

$$v = \sum_{i=0}^p w_i x_i = \mathbf{w}^\top \mathbf{x}.$$

# Redes Neuronais

- Atualmente, a RN mais simples consiste de entradas, de uma camada intermediária escondida e das saídas.
- Sejam  $\mathbf{X} = (X_1, \dots, X_p)^\top$ ,  $\mathbf{Y} = (Y_1, \dots, Y_M)^\top$  e  $\mathbf{W} = (W_1, \dots, W_K)^\top$  e sejam, os vetores de pesos  $\alpha_j$ ,  $j = 1, \dots, M$ ,  $\beta_k$ ,  $k = 1, \dots, K$ , de ordens  $p \times 1$  e  $M \times 1$ , respectivamente.

Essa rede neural simples pode ser representada pelas equações:

$$Y_j = f(\alpha_{0j} + \alpha_j^\top \mathbf{X}), \quad j = 1, \dots, M, \quad (1)$$

$$W_k = \beta_{0k} + \beta_k^\top \mathbf{Y}, \quad k = 1, \dots, K, \quad (2)$$

$$f_k(\mathbf{X}) = g_k(\mathbf{W}), \quad k = 1, \dots, K. \quad (3)$$

- A função  $f$  é chamada **função de ativação** e geralmente é usada a sigmóide  $f(x) = 1/(1 + e^{-x})$ .  
Os pesos  $\alpha_{0j}$  e  $\beta_{0k}$  têm o mesmo papel de  $b$  no perceptron e representam vieses. A saída final é  $g_k(\mathbf{W})$ . Em problemas de regressão,  $g_k(\mathbf{W}) = W_k$  e em problemas de classificação,  $g_k(\mathbf{W}) = e^{W_k} / \sum_i e^{W_i}$ , que corresponde a uma logística multidimensional.

Os  $Y_j$  constituem a camada escondida e não são observáveis.

# Redes Neuronais

- O ajuste de modelos de RN é feito minimizando a soma dos quadrados dos resíduos, no caso de regressão, onde a minimização é sobre os pesos. No caso de classificação, usamos a taxa de erros de classificação. Nos dois casos é usado um algoritmo chamado de **backpropagation**. É necessário escolher valores iniciais e regularização (usando uma função penalizadora), porque o algoritmo de otimização é não convexo e instável.
- No caso de termos várias camadas intermediárias obtém-se o que é chamado aprendizado profundo (**deep learning**). A complexidade do algoritmo é proporcional ao número de observações, número de preditores, número de camadas e número de épocas de treinamento. Para detalhes sobre esses tópicos, veja Hastie et al. (2017) e Cholet (2018).
- Leo Breiman (2001) distingue dois paradigmas em modelagem estatística: **data model** e **algorithmic model**, onde o segundo engloba os algoritmos usados em ML. Segundo ele, a maioria dos métodos importantes estão na categoria 2.

## Redes Neuronais: Perceptron

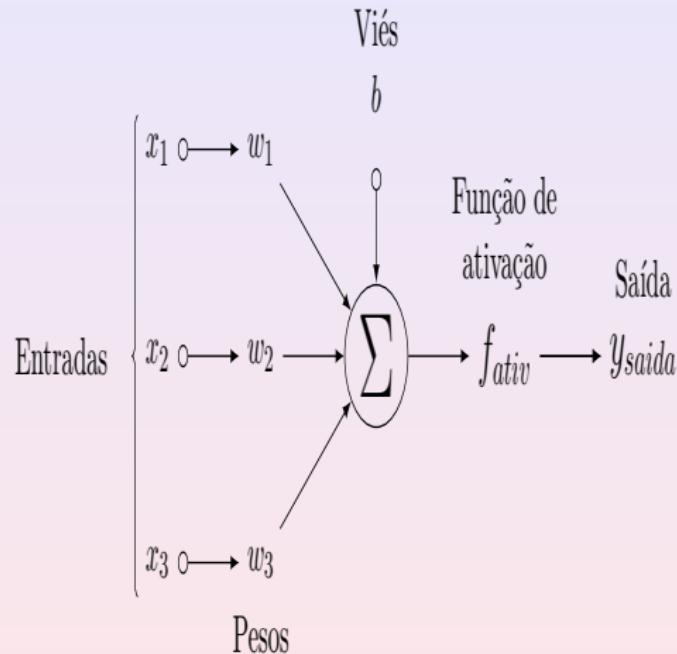
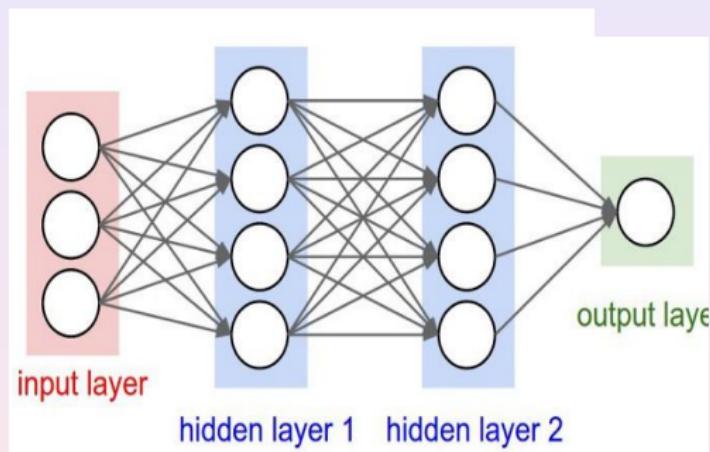


Figura: Perceptron de Rosenblatt

# Redes Neuronais



## Referências

- Breiman, L. (2001). Statistical modeling: the two cultures. *Statistical Science*, **16**, 199–231.
- Chambers, J. M. (1993). Greater or lesser Statistics: A choice for future research. *Statistics and Computing*, **3**, 182–184.
- Chollet, F. (2018). *Deep Learning with R*. Manning.
- Cleveland, W. M. (1985). *The Elements of Graphing Data*. Monterey: Wadsworth.
- Cleveland, W. M. (1993). *Visualizing Data*. Summit, New Jersey: Hobart Press.
- Cleveland, W. M. (2001). Data Science: An action plan for expanding the technical areas of the field of Statistics. *International Statistical Review*, **69**, 21–26.
- Donoho, D. (2017). 50 years of Data Science. *Journal of Computational and Graphical Statistics*, **26**, 745–766.

## Referências

- Hastie, T., Tibshirani, R. and Friedman, J. (2017). *The Elements of Statistical Learning*, 2nd Edition, Springer.
- Hebb, D. O. (1949). *The organization of behavior*. New York: Wiley.
- Jordan, M. I. (2019). Artificial intelligence – The revolution hasn't happened yet. *Harvard Data Science Review*, Issue 1.1.
- McCulloch, W. S. and Pitts, W. A. (1943). Logical calculus of the ideas immanent in nervous activity. *Butt. math. Biophysics*, S, 115–133.
- Rosenblatt, F. (1958). The perceptron: A theory of statistical separability in cognitive systems. Buffalo: Cornell Aeronautical Laboratory, Inc. Rep. No. VG-1196-G-1.
- Tukey, J. W. (1962). The future of data analysis. *The Annals of Mathematical Statistics*, **33**, 1–67.
- Tukey, J. W. (1977). *Exploratory Data Analysis*. Reading: Addison-Wesley.
- Turing, A. (1950). Computing machinery and intelligence". *Mind*, LIX (236).

# MAE 5905: Introdução à Ciência de Dados

Pedro A. Morettin

Instituto de Matemática e Estatística  
Universidade de São Paulo  
[pam@ime.usp.br](mailto:pam@ime.usp.br)  
<http://www.ime.usp.br/~pam>

## Aula 3

20 de março de 2023

# Sumário

1 Big Data

2 Modelos para o AE

3 Métodos de estimação

## Notação

1. Matriz de dados  $\mathbf{X}$ , de ordem  $n \times p$ ;  $n$  amostras (indivíduos),  $p$  variáveis.

$$\mathbf{X} = [x_{ij}] = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ x_{21} & x_{22} & \cdots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{np} \end{bmatrix}.$$

2. linhas de  $\mathbf{X}$ :  $x_1, \dots, x_n$ ; cada  $x_i$  é um vetor  $p \times 1$ ;

colunas de  $\mathbf{X}$ :  $\mathbf{x}_1, \dots, \mathbf{x}_p$ ; cada  $\mathbf{x}_j$  é um vetor  $n \times 1$ .

3. Podemos escrever

$$\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_p] = \begin{bmatrix} \mathbf{x}_1^\top \\ \mathbf{x}_2^\top \\ \vdots \\ \vdots \\ \mathbf{x}_n^\top \end{bmatrix}.$$

4.  $y_i$ =i-ésima observação,  $\mathbf{y} = (y_1, \dots, y_n)^\top$ . No caso de AE supervisionado,  $y_i$  é resposta aos preditores  $x_i$ , num problema de regressão, e é o rótulo da i-ésima classe, num problema de classificação.

Dados:  $\{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$ , cada  $x_i$  um vetor  $p \times 1$ .

# Tipos de dados

- grande número de amostras,  $n$ , e pequeno número de variáveis,  $p$ ;
- pequeno número de amostras,  $n$ , e grande número de variáveis,  $p$ ;
- grande número de amostras,  $n$ , e grande número de variáveis,  $p$ .
- **Dados de Alta Dimensão:**  $n < p$ .
- SAS ([www.sas.com/en\\_us/insightsbig-datawhat-is-big-data.html](http://www.sas.com/en_us/insightsbig-datawhat-is-big-data.html)): “Big data is a term that describes a large volume of data - both **structured** and **unstructured** - that inundates a business on a day-to-day basis. But it is not the amount of data that's important. It's what organizations do with the data that matters. Big data can be analyzed for insights that lead to better decisions and strategic business moves.”

## Dados de alta dimensão

- **Alta dimensão relativa:** modelos com muita variáveis ( $p$ ) comparado com o número de amostras ( $n$ ), mas usualmente com  $p < n$ ;
- **Alta dimensão moderada:** modelos com número de variáveis proporcional ao número de amostras, usualmente  $p > n$ ;
- **Alta dimensão:** modelos com mais variáveis do que amostras, e o número de variáveis cresce polinomialmente ou exponencialmente com  $n$ .

# Tipos de dados

- **Dados estruturados:** informação organizada que se ajusta a estruturas usuais de bases de dados, relativamente fáceis de armazenar e analisar. Exemplos usuais de dados numéricos ou não, que podem ser dispostos em uma matriz de dados.
- **Dados não estruturados:** tudo que não se encaixa no item anterior, como arquivos de textos, páginas da web, email, mídias sociais etc.
- Os quatro V's dos Big Data: **VOLUME** (escala dos dados); **VARIEDADE** (formas diferentes de dados); **Velocidade** (análise de *streaming data*); **VERACIDADE** (incerteza sobre os dados). ([www.ibmbigdatahub.com](http://www.ibmbigdatahub.com))

# Tipos de dados

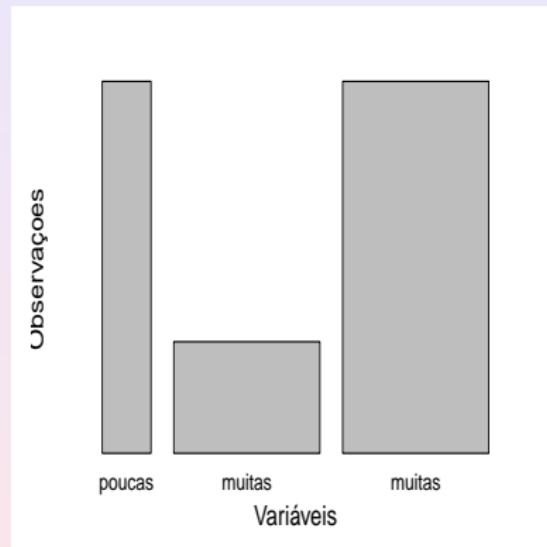


Figura 1: Tipos de dados

## AE supervisionado

- Big Data implica em Big Models
- Big model: número grande de parâmetros ( $p$ ) a serem estimados por algum método estatístico, comparado com o número de observações ( $n$ ).
- Exemplo: regressão linear

$$y_i = \beta_0 + \beta_1 x_{1i} + \dots + \beta_p x_{pi} + \varepsilon_i, \quad i = 1, \dots, n. \quad (1)$$

- Problema se  $p \gg n$  (dados de alta dimensão)
- Estimação em modelos lineares com dados de alta dimensão pode ser tratada:
  - (a) usando técnicas de redução da dimensão, como por exemplo ACP, AF e ACI;
  - (b) usando estimação penalizada (regularização);
  - (c) métodos bayesianos;
- No caso de modelos não lineares: árvores de decisão, redes neurais, bagging, boosting.

## AE supervisionado: Exemplo 1

- Se quisermos saber se há relação entre o consumo privado (variável  $C$ ) e renda disponível (variável  $Y$ ) de indivíduos de uma população, podemos escolher uma amostra de  $n$  indivíduos dessa população e medir essas duas variáveis nesses indivíduos, obtendo-se o conjunto de dados  $\{(Y_1, C_1), \dots, (Y_n, C_n)\}$ .
- Em Economia, sabe-se, desde Keynes, que o gasto com o consumo de pessoas ( $C$ ) é uma função da renda pessoal disponível ( $Y$ ), ou seja

$$C = f(Y),$$

para alguma função  $f$ .

- Para se ter uma ideia de como é a função  $f$  para essa comunidade, podemos construir um gráfico de dispersão entre  $Y$  e  $C$ . Com base em um conjunto de dados hipotéticos com  $n = 20$ , esse gráfico está apresentado na Figura 1 e é razoável postular o modelo

$$C_i = \alpha + \beta Y_i + \varepsilon_i, \quad i = 1, \dots, n, \tag{2}$$

em que  $(Y_i, C_i)$ ,  $i = 1, \dots, n$  são os valores de  $Y$  e  $C$  efetivamente observados e  $\varepsilon_i$ ,  $i = 1, \dots, n$  são variáveis não observadas, chamadas **erros**. Aqui,  $n = 20$  e  $p = 1$ .

## AE supervisionado: Exemplo 1

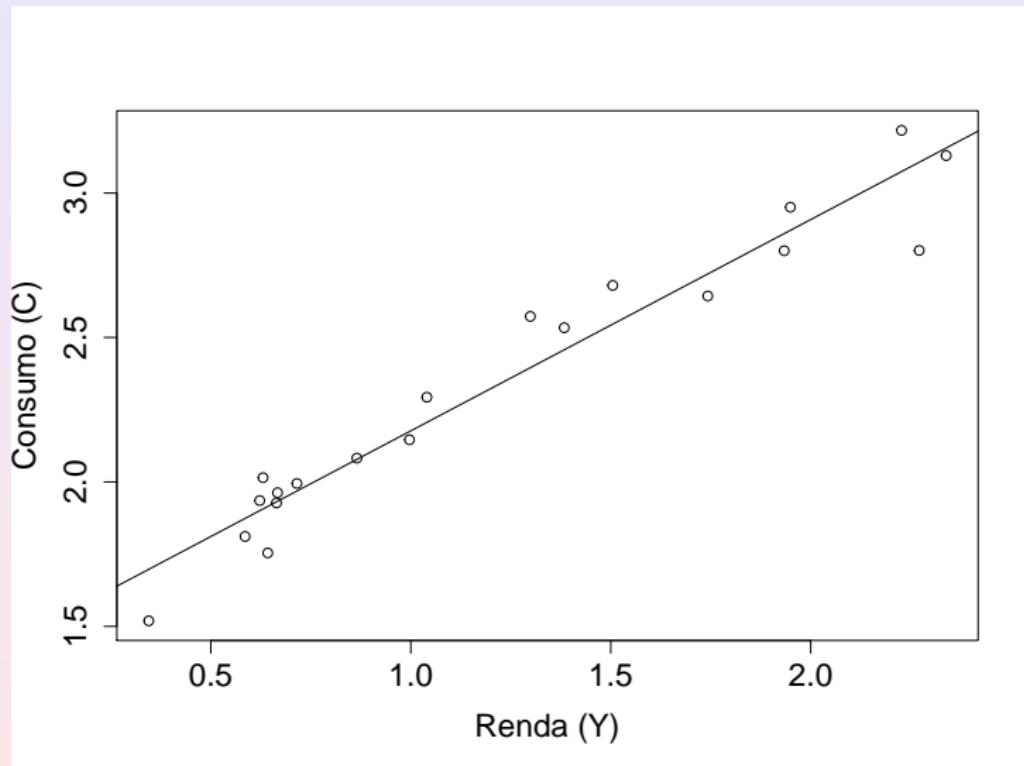


Figura 1: Relação entre renda e consumo de 20 indivíduos.

## AE supervisionado: Exemplo 2

- Os dados apresentados na planilha **Esforço** são provenientes de um estudo sobre teste de esforço cardiopulmonar em pacientes com insuficiência cardíaca realizado no InCor da Faculdade de Medicina da USP. Um dos objetivos do estudo é comparar os grupos formados pelas diferentes etiologias quanto às respostas respiratórias e metabólicas obtidas do teste de esforço cardiopulmonar. Outro objetivo do estudo é saber se alguma das características observadas (ou combinação delas) pode ser utilizada como fator prognóstico de óbito.
- Nosso objetivo poderia ser desenvolver um modelo que possa ser usado para prever o consumo de oxigênio **VO2** (variável resposta,  $y$ ) com as informações sobre **Carga** na esteira ergométrica ( $x_1$ ) e **IMC** ( $x_2$ ) (preditores). Um modelo de regressão linear seria

$$y_i = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \varepsilon_i, \quad i = 1, \dots, 127. \quad (3)$$

Aqui,  $p = 2$  e teremos que estimar  $\beta_0, \beta_1, \beta_2$  mais os parâmetros associados à variável  $\varepsilon_i$ . A Figura 2 mostra que o modelo (plano) não parece ser adequado, há vários pontos bastante afastados do plano.

## AE supervisionado: Exemplo 2

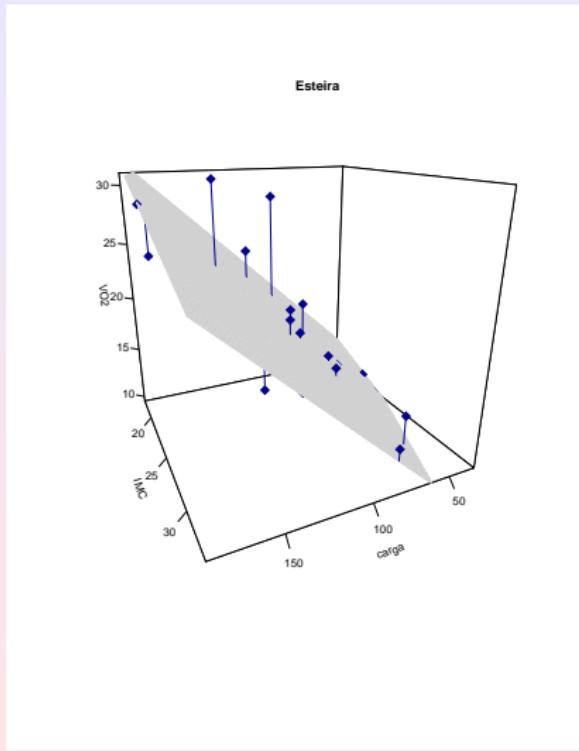


Figura 2: Consumo de oxigênio como função de Carga e IMC.

## AE supervisionado: regressão

- Dados o vetor  $y$  de variáveis respostas e os preditores  $x_i$ , o modelo geral é da forma

$$y_i = f(x_i) + \varepsilon_i, \quad i = 1, \dots, n, \tag{4}$$

com  $E(\varepsilon_i) = 0$ ,  $\varepsilon_i$  ortogonal a  $x_i$  e  $f$  desconhecida, chamada de **informação sistemática**.

- O objetivo do AE é encontrar métodos para estimar  $f$ .
- Dois motivos para estimar  $f$ : **Previsão** e **Inferência**

## Previsão

- Obtido o estimador  $\hat{f}$ , obtemos o previsor de  $\mathbf{y}$

$$\hat{\mathbf{y}} = \hat{f}(\mathbf{X}).$$

- A acurácia de  $\hat{\mathbf{y}}$  como previsor de  $\mathbf{y}$  depende de dois erros:

**erro redutível**: introduzido pelo estimador de  $f$ ; assim chamado porque podemos melhorar a acurácia de  $\hat{f}$  usando uma técnica de AE mais apropriada;

**erro irredutível**: mesmo usando o melhor estimador de  $f$ , esse erro depende de  $\varepsilon$ , que não pode ser previsto usando  $\mathbf{X}$ .

- Supondo  $\hat{f}$  e  $\mathbf{X}$  fixos, pode-se ver que

$$\begin{aligned} E(\mathbf{y} - \hat{\mathbf{y}})^2 &= E[f(\mathbf{X}) + \varepsilon - \hat{f}(\mathbf{X})]^2 \\ &= E[f(\mathbf{X}) - \hat{f}(\mathbf{X})]^2 + \text{Var}(\varepsilon). \end{aligned} \tag{5}$$

O primeiro termo do segundo membro representa o erro redutível e, o segundo termo, o erro irredutível. O objetivo é minimizar o erro redutível.

# Inferência

O interesse pode não ser fazer previsões, mas entender como a resposta é afetada pela variação dos preditores.

Interesse nos seguintes tópicos:

- Identificar alguns preditores importantes, dentre todos os preditores;
- Relação entre a resposta e cada um dos preditores; no Exemplo 2,  $y$  cresce com a carga, mas decresce com o IMC;
- A relação entre a resposta e cada preditor é linear, ou mais complicada? Modelos lineares fornecem interpretações mais simples, mas em geral previsões menos acuradas. Modelos não lineares fornecem previsões mais acuradas, mas o modelo perde em interpretabilidade.

## Métodos paramétricos

- Fazemos alguma suposição sobre a forma de  $f$ , por exemplo, o modelo de regressão dado em (1). O problema simplifica-se, pois temos que estimar  $p + 1$  parâmetros.
- Selecionado o modelo, temos que ajustá-lo aos dados de treinamento (*treinar* o modelo). No caso do modelo (1), o método mais usado é Mínimos Quadrados (MQ). Mas há outros métodos, como SVM.
- O ajuste do modelo (1) por MQ pode ser pobre, como no Exemplo 2 (ver Figura 2).
- Nesse caso, pode-se tentar modelos mais flexíveis, escolhendo-se outras formas funcionais para  $f$ , incluindo-se modelos não lineares.
- Modelos mais flexíveis envolvem estimar um número muito grande de parâmetros, aparecendo o problema do super-ajustamento (*overfitting*).

## Métodos não paramétricos

- Nesse caso, não fazemos nenhuma hipótese sobre a forma funcional de  $f$ .
- Como o problema não se reduz a estimar um número pequeno de parâmetros, necessitaremos de um número grande de observações para obter estimadores acurados de  $f$ .
- Vários métodos podem ser usados:
  - ◊ usando kernels
  - ◊ usando polinômios locais (e.g Lowess)
  - ◊ usando splines
  - ◊ usando polinômios ortogonais (e.g. Chebyshev )
  - ◊ usando outras bases ortogonais (e.g. Fourier, ondaletas)

## Acurácia e interpretação

- métodos menos flexíveis (e.g regressão linear) (ou mais restritivos) em geral são menos acurados e mais fáceis de interpretar.
- métodos mais flexíveis (e.g splines) são mais acurados e mais difíceis de interpretar.
- Para cada conjunto de dados, um método pode ser preferível a outros.
- Escolha do método é a parte mais difícil do AE/ML.

## Qualidade do ajuste

- No caso de regressão, a medida de ajuste mais usada é o Erro Quadrático Médio (EQM), dado por

$$\text{EQM} = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{f}(x_i))^2, \quad (6)$$

onde  $\hat{f}(x_i)$  é o preditor de  $y$  para a  $i$ -ésima observação.

- O EQM acima é calculado no conjunto de treinamento que produz  $\hat{f}$ . É chamado *EQM de treinamento*.
- Todavia, estamos mais interessados na acurácia do ajuste para os dados de teste.
- Se tivermos um grande número de dados de teste, poderemos calcular o *EQM de teste*,

$$\text{Média}(y_0 - \hat{f}(x_0))^2, \quad (7)$$

que é o erro de previsão quadrático médio para as observações teste  $(x_0, y_0)$ .

- Se não tivermos observações teste use (6). Na maioria dos casos o EQM de treinamento é menor que o EQM de teste.
- Para calcular o EQM de treinamento, usa-se CV.

## Viés/variância

- Para um dado  $(\mathbf{x}_0, y_0)$ ,

$$E[y_0 - \hat{f}(\mathbf{x}_0)]^2 = \text{Var}[\hat{f}(\mathbf{x}_0)] + [\text{Vies}(\hat{f}(\mathbf{x}_0))]^2 + \text{Var}(\varepsilon). \quad (8)$$

- Para minimizar (8), selecionamos um método que simultaneamente tiver baixo viés e baixa variância. O EQM de teste, em geral, apresenta uma forma de U, resultante da competição entre viés e variância.
- Métodos de AE mais flexíveis têm viés baixo e variância grande.
- Na prática,  $f$  não é conhecido e não é possível calcular o EQM de teste, viés e variância para um método de AE.

# Classificação

- No caso de respostas  $y_1, \dots, y_n$  qualitativas temos um problema de **classificação**.
- Formalmente, seja  $(\mathbf{x}, y)$ , de modo que  $\mathbf{x} \in \mathbb{R}^p$  e  $y \in \{-1, 1\}$ . Então, um **classificador** é uma função  $g : \mathbb{R}^p \rightarrow \{-1, 1\}$  e a **função erro** ou **risco** é a probabilidade de erro,  $L(g) = P\{g(\mathbf{X}) \neq Y\}$ .
- Obtendo-se um estimador de  $g$ , digamos  $\hat{g}$ , sua acurácia pode ser medida pelo estimador de  $L(g)$ , chamado de **taxa de erro de treinamento**, que é a proporção de erros gerados pela aplicação de  $\hat{g}$  às observações de treinamento, ou seja,

$$\hat{L}(\hat{g}) = \frac{1}{n} \sum_{i=1}^n I(y_i \neq \hat{y}_i), \quad (9)$$

sendo que  $\hat{y}_i = \hat{g}(x_i)$  é o rótulo (-1 ou 1) da classe prevista usando  $\hat{g}$ .

## Classificador de Bayes

- O interesse está na **taxa de erro de teste**

$$\text{Média}(I(y_0 \neq \hat{y}_0)), \quad (10)$$

para observações de teste  $(\mathbf{x}_0, y_0)$ . Um bom classificador tem (10) pequeno.

- Pode-se provar que (10) é minimizado, em média, por um classificador que associa cada observação à classe mais provável, dados os preditores; ou seja, temos que maximizar

$$P(y = j | \mathbf{x} = \mathbf{x}_0). \quad (11)$$

Tal classificador é chamado de Bayes.

- No caso de duas classes, classificar na classe -1 se  $P(y = -1 | \mathbf{x} = \mathbf{x}_0) > 0,5$  e na classe 1 c.c. O classificador de Bayes produz a menor taxa de erro e será dada por  $1 - \max_j P(y = j | \mathbf{x} = \mathbf{x}_0)$ . A taxa de erro de Bayes global é dada por  $1 - E(\max_j P(y = j | \mathbf{x} = \mathbf{x}_0))$ , onde  $E(\cdot)$  é caculada sobre todos os valores de  $\mathbf{x}$ .

## *K*-ésimo vizinho mais próximo

- O classificador de Bayes não pode ser calculado na prática, pois não conhecemos a distribuição condicional de  $y$  dado  $\mathbf{x}$ .
- Uma possibilidade é estimar a distribuição condicional e, então, estimar (11).
- O classificador *K*-ésimo vizinho mais próximo (*K*-nearest neighbors, KNN) estima tal distribuição da seguinte maneira:
  - (i) Escolha  $K > 0$  inteiro e uma observação teste  $\mathbf{x}_0$ .
  - (ii) O classificador KNN primeiro identifica os  $K$  pontos do conjunto de treinamento mais próximos de  $\mathbf{x}_0$ ; chame-os de  $\mathcal{N}$ .
  - (iii) Estime a probabilidade condicional da classe  $j$  por

$$P(y = j | \mathbf{x} = \mathbf{x}_0) = \frac{1}{K} \sum_{i \in \mathcal{N}} I(y_i = j). \quad (12)$$

- (iv) Classifique  $\mathbf{x}_0$  na classe com a maior probabilidade condicional.
- Escolha de  $K$  crucial; resultado depende dessa escolha.

## Modelos para séries temporais

- (1) Consideremos uma série temporal multivariada  $\mathbf{z}_t, t = 1, \dots, T$  em que  $\mathbf{z}_t$  contém valores de  $d$  variáveis e uma série temporal univariada  $Y_t, t = 1, \dots, T$ . O objetivo é fazer previsões de  $Y_t$ , para horizontes  $h = 1, \dots, H$ , com base nos valores passados de  $Y_t$  e de  $\mathbf{z}_t$ . Uma suposição básica é que o processo  $\{Y_t, \mathbf{z}_t\}, t \geq 1$  seja estacionário fraco (ou de segunda ordem), veja Morettin (2017), com valores em  $\mathbb{R}^{d+1}$ . Para  $p \geq 1$ , consideramos o processo vetorial  $n$ -dimensional

$$\mathbf{x}_t = (Y_{t-1}, \dots, Y_{t-p}, \mathbf{z}_t^\top, \dots, \mathbf{z}_{t-r}^\top)^\top, \quad \text{com } n = p + d(r + 1).$$

- (2) O modelo a considerar é uma extensão do modelo (4), ou seja,

$$Y_t = f(\mathbf{x}_t) + e_t, \quad t = 1, \dots, T. \tag{13}$$

# O modelo geral

- (3) Como em (4),  $f$  é uma função desconhecida e  $e_t$  tem média zero e variância finita. O objetivo é estimar  $f$  e usar o modelo para fazer previsões para um horizonte  $h$  por meio de

$$Y_{t+h} = f_h(\mathbf{x}_t) + e_{t+h}, \quad h = 1, \dots, H, \quad t = 1, \dots, T.$$

Para uma avaliação do método de previsão, a acurácia é

$$\Delta_h(\mathbf{x}_t) = |\hat{f}_h(\mathbf{x}_t) - f_h(\mathbf{x}_t)|.$$

- (4) Em geral, a norma  $L_q$  é  $E\{|\Delta_h(\mathbf{x}_t)|^q\}^q$ , sendo que as mais comumente usadas consideram  $q = 1$  (correspondente ao erro de previsão absoluto médio) ou  $q = 2$  (correspondente ao erro de previsão quadrático médio). A raiz quadrada dessas medidas também é utilizada.  
Escolhendo-se uma função perda, o objetivo é selecionar  $f_h$  a partir de um conjunto de modelos que minimize o risco, ou seja, o valor esperado da norma  $L_q$ .

# Modelos lineares

- (1) Modelos frequentemente usados têm a forma (13) em que  $f(\mathbf{x}) = \boldsymbol{\beta}^\top \mathbf{x}$ , com  $\boldsymbol{\beta}$  denotando um vetor de  $\mathbb{R}^n$ . Estimadores de mínimos quadrados não são únicos se  $n > T$ . A ideia é considerar modelos lineares com alguma função de penalização, ou seja que minimizem

$$Q(\boldsymbol{\beta}) = \sum_{t=1}^{T-h} (Y_{t+h} - \boldsymbol{\beta}^\top \mathbf{x}_t)^2 + p(\boldsymbol{\beta}),$$

em que  $p(\boldsymbol{\beta})$  depende, além de  $\boldsymbol{\beta}$ , de  $\mathbf{Z}_t$ , de um parâmetro de suavização  $\lambda$  e de eventuais hiperparâmetros.

- (2) Este processo denomina-se **regularização**. Há várias formas de regularização como aquelas conhecidas por *Ridge*, *Lasso*, *Elastic Net* e generalizações. Detalhes sobre esses processos de regularização são apresentados no Capítulo 8.

# Modelos não lineares

- (1) Num contexto mais geral, o objetivo é minimizar

$$S(f) = \sum_{t=1}^{T-h} [Y_{t+h} - f(\mathbf{x}_t)]^2,$$

para  $f$  pertencendo a algum espaço de funções  $\mathcal{H}$ .

- (2) Por exemplo, podemos tomar  $f$  como uma função contínua, com derivadas contínuas, ou  $f$  apresentando alguma forma de descontinuidade etc. Esses espaços, são, em geral, de dimensão infinita e a solução pode ser complicada. Para contornar esse problema, pode-se considerar uma coleção de espaços de dimensão finita  $\mathcal{H}_d$ , para  $d = 1, 2, \dots$ , de tal sorte que  $\mathcal{H}_d$  converja para  $\mathcal{H}$  segundo alguma norma. Esses espaços são denominados **espaços peneira** (*sieve spaces*).
- (3) Para cada  $d$ , consideramos a aproximação

$$h_d(\mathbf{x}_t) = \sum_{j=1}^J \beta_j h_j(\mathbf{x}_t),$$

em que  $h_j(\cdot)$  são **funções base** para  $\mathcal{H}_d$  e tanto  $J$  como  $d$  são funções de  $T$ . Podemos usar *splines*, polinômios, funções trigonométricas, ondaletas etc. como funções base.

## Modelos não lineares

- (4) Se as funções base são conhecidas, elas são chamadas *linear sieves* e se elas dependem de parâmetros a estimar, são chamadas *nonlinear sieves*.
- (5) Exemplos de non linear sieves são as árvores de decisão e as redes neurais.
- (6) Métodos em AE ou ML são dedicados a observações de variáveis independentes e identicamente distribuídas. O caso de séries temporais é mais complicado e foge ao escopo deste curso. Apresentaremos apenas algumas ideias relacionadas a esse tópico.

## Referências

- James, G., Witten, D., Hastie, T. and Tibshirani, R. (2017). *An Introduction to Statistical Learning*. Springer.
- Morettin, P. A. e Singer, J. M. (2022). *Estatística e Ciência de Dados*. LTC.

# MAE 5905: Introdução à Ciência de Dados

Pedro A. Morettin

Instituto de Matemática e Estatística  
Universidade de São Paulo  
pam@ime.usp.br  
<http://www.ime.usp.br/~pam>

## Aula 4

27 de março de 2023

# Sumário

## 1 Regressão Linear Simples

# Modelos de regressão linear

1. Um dos modelos estatísticos mais usados na prática: **modelo de regressão**.
2. Exemplo mais simples: dados  $(x_1, y_1), \dots, (x_n, y_n)$  de duas variáveis contínuas  $X$  e  $Y$  num contexto em que sabemos a priori que a distribuição de probabilidades de  $Y$  pode depender de  $X$ , ou seja,  $X$  é a variável explicativa (ou preditora) e  $Y$  é a variável resposta.
3. Trata-se de um **modelo linear**, ou seja, os parâmetros aparecem no modelo de forma linear.
4. Trata-se, também, de um **modelo paramétrico**.

# Regressão linear simples

- **Exemplo:** objetivo é avaliar como a distância com que indivíduos conseguem distinguir um determinado objeto (doravante indicada simplesmente como distância) varia com a idade.
- Figura 1: gráfico de dispersão: tendência decrescente da distância com idade.
- modelo:  $y_i = \alpha + \beta x_i + e_i, \quad i = 1, \dots, n$
- $\alpha, \beta$  **parâmetros**
- modelo: **regressão linear simples – RLS**
- mais adequado:  $y_i = \alpha + \beta(x_i - 18) + e_i, \quad i = 1, \dots, n.$

# Regressão linear simples

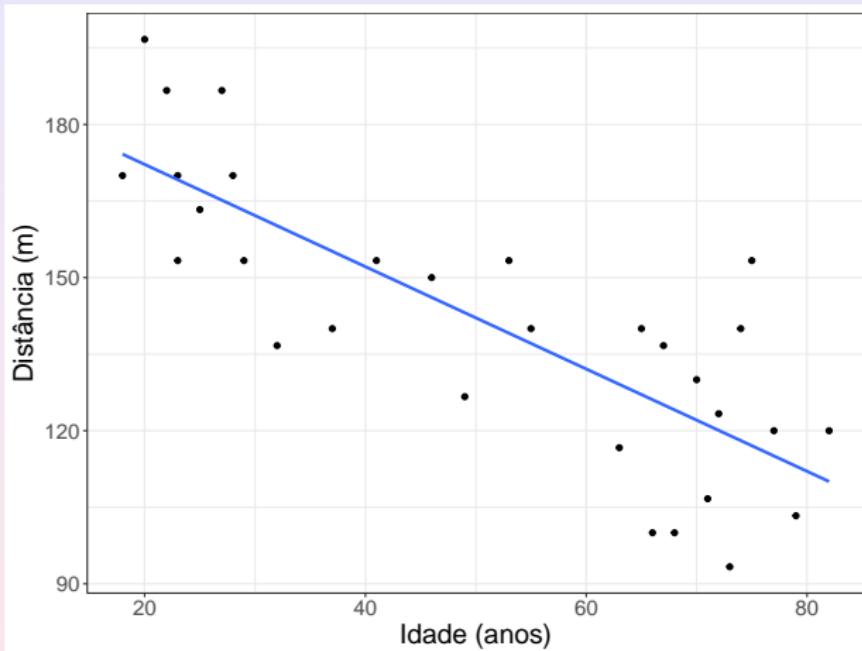


Figura 1: Gráfico de dispersão para os dados **distância**

# Estimação do modelo RLS

- SQE:  $Q(\alpha, \beta) = \sum_{i=1}^n e_i^2 = \sum_{i=1}^n (y_i - \alpha - \beta x_i)^2$ .
- Estimadores de mínimos quadrados (EMQ): minimizam a SQE.

- 

$$\hat{\beta} = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2}, \quad (1)$$

e

$$\hat{\alpha} = \bar{y} - \hat{\beta} \bar{x} \quad (2)$$

em que  $\bar{x} = n^{-1} \sum_{i=1}^n x_i$  e  $\bar{y} = n^{-1} \sum_{i=1}^n y_i$

- Um estimador de  $\sigma^2$  é

$$S^2 = \frac{1}{n-2} Q(\hat{\alpha}, \hat{\beta}) = \frac{1}{n-2} \sum_{i=1}^n \hat{e}_i^2 = \sum_{i=1}^n (y_i - \hat{\alpha} - \hat{\beta} x_i)^2, \quad (3)$$

em que onde  $Q(\hat{\alpha}, \hat{\beta})$  é a **soma dos quadrados dos resíduos**, abreviadamente, **SQRRes**.

# Uso do R para o ajuste

- função lm() do pacote MASS
- modelo:  $distancia_i = \alpha + \beta(idade_i - 18) + e_i, \quad i = 1, \dots, n$   
 $\hat{y}_i = 174.23 - 1.004(x_i - 18)$ .
- modelo:  $distancia_i = \alpha + \beta idade_i + e_i, \quad i = 1, \dots, n$   
 $\hat{y}_i = 192.3 - 1.004x_i$ ,
- Residual standard error: 16.6 on 28 degrees of freedom
- Multiple R-squared: 0.6424, Adjusted R-squared: 0.6296

# RLS-Avaliação do ajuste

- Uma vez ajustado o modelo, convém avaliar a qualidade do ajuste e um dos indicadores mais utilizados para essa finalidade é o **coeficiente de determinação** definido como

$$R^2 = \frac{SQTot - SQRes}{SQTot} = \frac{SQReg}{SQTot} = 1 - \frac{SQRes}{SQTot}$$

em que a soma de quadrados total é  $SQTot = \sum_{i=1}^n (y_i - \bar{y})^2$ , a soma de quadrados dos resíduos é  $SQRes = \sum_{i=1}^n (y_i - \hat{y}_i)^2$  e a soma de quadrados da regressão é  $SQReg = \sum_{i=1}^n (\hat{y}_i - \bar{y})^2$ .

- Esse coeficiente mede a porcentagem da variação total dos dados (em relação à sua média) explicada pelo modelo de regressão. O coeficiente de determinação deve ser acompanhado de outras ferramentas para a avaliação do ajuste, pois não está direcionado para identificar se todas as suposições do modelo são compatíveis com os dados sob investigação. Em particular, mencionamos os gráficos de resíduos, gráficos de Cook e gráficos de influência local.

## $R^2$ ajustado

- Toda a vez que incluimos um preditor ao modelo, o  $R^2$  aumenta; nunca decresce. Consequentemente, um modelo com mais parâmetros pode parecer que tenha o melhor ajuste (sobreajuste).
- O  $R^2$  ajustado cresce somente quando o preditor incluído realmente aumenta o poder preditivo do modelo de regressão.
- O  $R^2$  ajustado é calculado por

$$\bar{R}^2 = 1 - (1 - R^2) \left[ \frac{n - 1}{n - (k + 1)} \right],$$

em que  $n$  é o tamanho da amostra e  $k$  o número de preditores do modelo.

## RLS-Gráfico de resíduos

O gráfico de resíduos correspondente ao modelo ajustado aos dados **distancia** está apresentado na Figura 2.

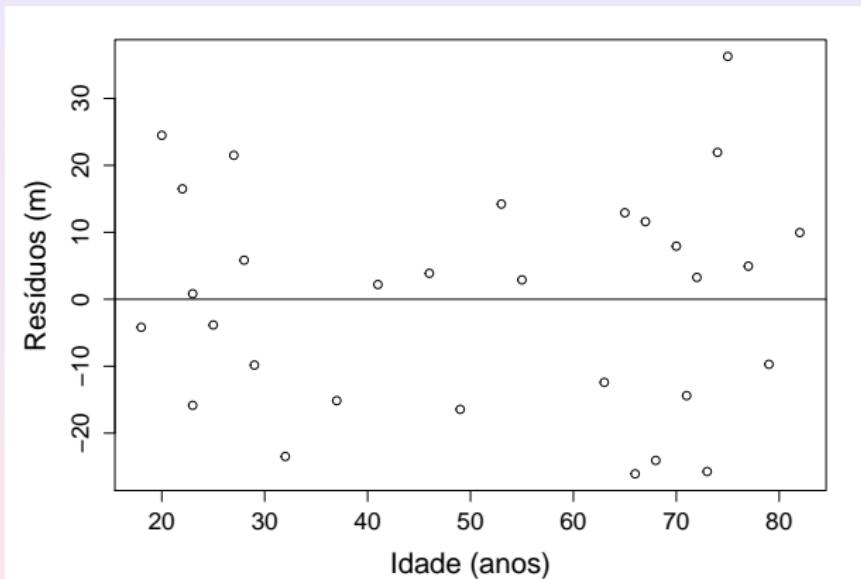


Figura 2: Gráfico de resíduos para o ajuste do modelo de regressão linear simples aos dados **distância**.

## RLS-resíduos padronizados

- Para facilitar a visualização em relação à dispersão dos resíduos e para efeito de comparação entre ajustes de modelos em que as variáveis resposta têm unidades de medida diferentes, convém padronizá-los, i.e., dividi-los pelo respectivo desvio padrão para que tenham variância igual a 1.
- Como os resíduos (ao contrário dos erros) são correlacionados, pode-se mostrar que

$$DP(\hat{e}_i) = \sigma \sqrt{1 - h_{ii}} \text{ com } h_{ii} = \frac{1}{n} + \frac{(x_i - \bar{x})^2}{\sum_{i=1}^n x_i^2 - n\bar{x}^2},$$

de forma que os **resíduos padronizados**, também chamados de **resíduos studentizados** são definidos por

$$\hat{e}_i^* = \hat{e}_i / (S \sqrt{1 - h_{ii}}) \quad (4)$$

- Os resíduos padronizados são adimensionais e têm variância igual a 1, independentemente da variância da variável resposta. Além disso, para erros com distribuição Normal, cerca de 99% dos resíduos padronizados têm valor entre -3 e +3.
- $h_{ii}$  : **leverage** (alavancagem) do  $i$ -ésimo preditor.

## RLS- Resíduos padronizados

O gráfico de resíduos padronizados correspondente àquele da Figura 2 está apresentado na Figura 3.

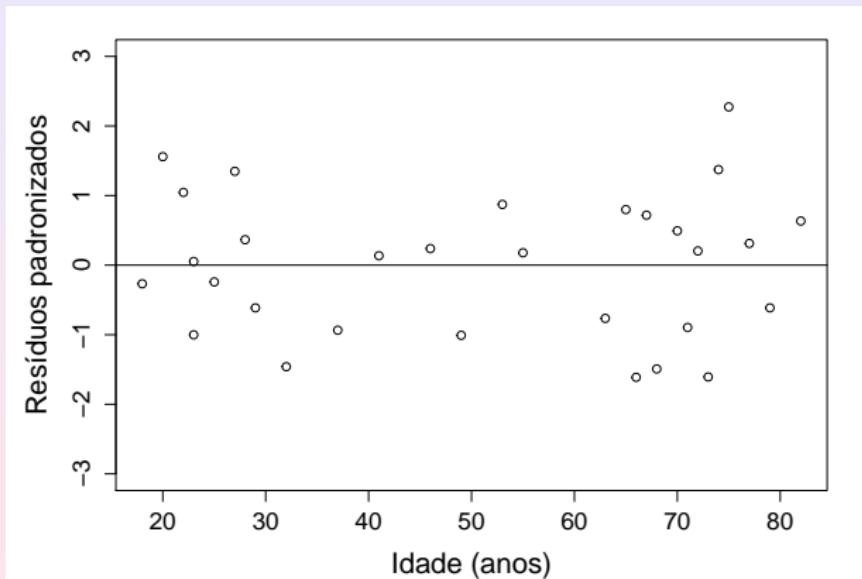


Figura 3: Gráfico de resíduos padronizados para o ajuste do modelo de regressão linear simples aos dados **distancia**.

# RLS-Distância de Cook

**Exemplo:** Consideremos agora os dados (hipotéticos) dispostos na Tabela 1, aos quais ajustamos um modelo de regressão linear simples.

Tabela 1: Dados hipotéticos

X	10	8	13	9	11	14	6	4	12	7	5	18
Y	8,04	6,95	7,58	8,81	8,33	9,96	7,24	4,26	10,84	4,82	5,68	6,31

## RLS-Distância de Cook

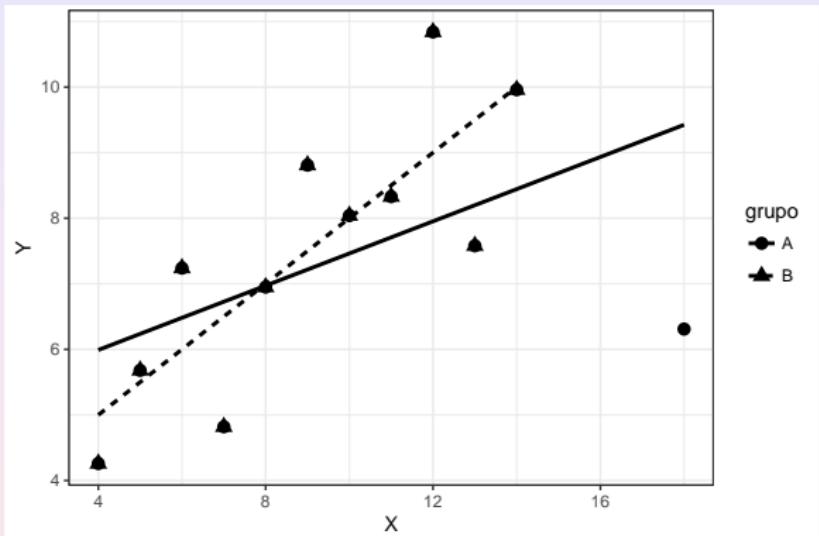


Figura 4: Gráfico de dispersão (com retas de regressão sobrepostas) para os dados da Tabela 1; curva sólida para dados completos e curva interrompida para dados com ponto influente eliminado.

## RLS-Distância de Cook

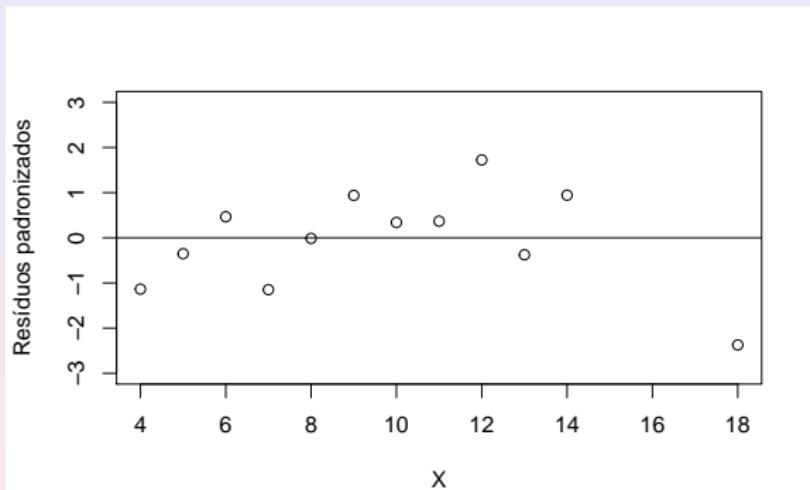


Figura 5: Gráfico de resíduos padronizados para o ajuste do modelo de regressão linear aos dados da Tabela 1.

# RLS-Distância de Cook

- A **distância de Cook** é uma maneira de identificar pontos influentes (**outliers**) em um conjunto de preditores, que afetam o modelo. É uma combinação da alavancagem de cada observação e dos resíduos. Quanto maior a alavancagem, maior é a distância de Cook.
- Denotando por  $\hat{\mathbf{y}}$  o vetor (de dimensão  $n$ ) com os valores preditos obtidos do ajuste do modelo baseado nas  $n$  observações e por  $\hat{\mathbf{y}}^{(-i)}$  o correspondente vetor com valores preditos (de dimensão  $n$ ) obtido do ajuste do modelo baseado nas  $n - 1$  observações restantes após a eliminação da  $i$ -ésima, a **distância de Cook** é definida como

$$D_i = \frac{(\hat{\mathbf{y}} - \hat{\mathbf{y}}^{(-i)})^\top (\hat{\mathbf{y}} - \hat{\mathbf{y}}^{(-i)})}{(p+1)S}$$

em que  $p$  é o número de coeficientes de regressão e  $S$  é uma estimativa do desvio padrão.

- Pode-se mostrar que a distância de Cook ( $D_i$ ) pode ser calculada sem a necessidade de ajustar o modelo com a omissão da  $i$ -ésima observação por meio da expressão

$$D_i = \frac{1}{p+1} \hat{e}_i^2 \frac{h_{ii}}{(1-h_{ii})^2},$$

lembrando que  $h_{ii}$  é a leverage.

## RLS-Distância de Cook

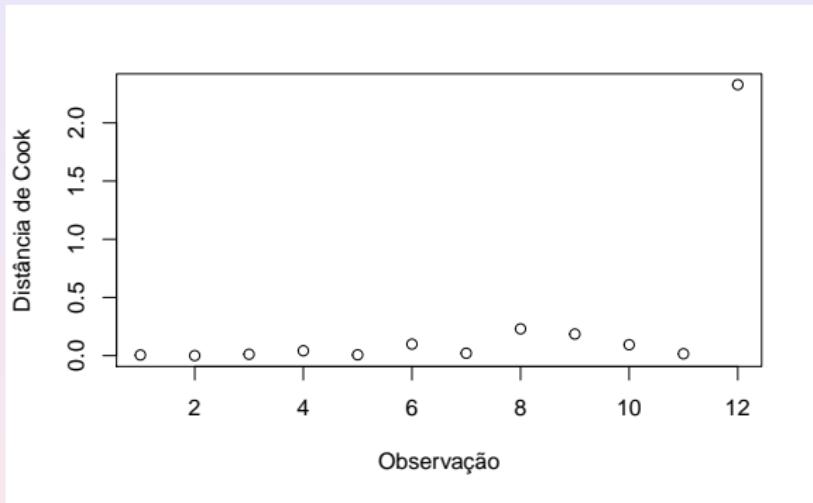


Figura 6: Gráfico de Cook correspondente ao ajuste do modelo de regressão linear aos dados da Tabela 1.

# RLS-Gráficos QQ

- Nos casos em que se supõe que os erros têm distribuição Normal, pode-se utilizar gráficos QQ (quantis–quantis) com o objetivo de avaliar se os dados são compatíveis com essa suposição. É importante lembrar que esses gráficos QQ devem ser construídos com os quantis amostrais baseados nos resíduos e não com as observações da variável resposta, pois apesar de suas distribuições também serem normais, suas médias variam com os valores associados da variável explicativa, ou seja, a média da variável resposta correspondente a  $y_i$  é  $\alpha + \beta x_i$ .
- Convém observar que sob normalidade dos erros, os resíduos padronizados seguem uma distribuição  $t$  com  $n - 2$  graus de liberdade e é dessa distribuição que se devem obter os quantis teóricos para a construção do gráfico QQ. Também deve-se lembrar que para valores de  $n$  maiores que 20 ou 30, os quantis da distribuição  $t$  se aproximam daqueles da distribuição Normal, tornando-as intercambiáveis para a construção do correspondente gráfico QQ.

## RLS-Gráficos QQ

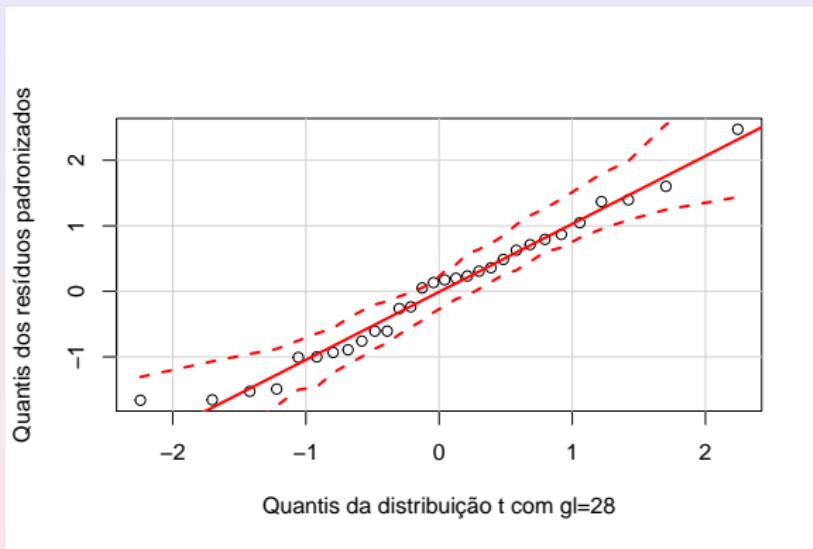


Figura 7: Gráfico QQ correspondente ajuste do modelo de regressão linear aos dados **distancia**.

## RLS-Gráficos QQ

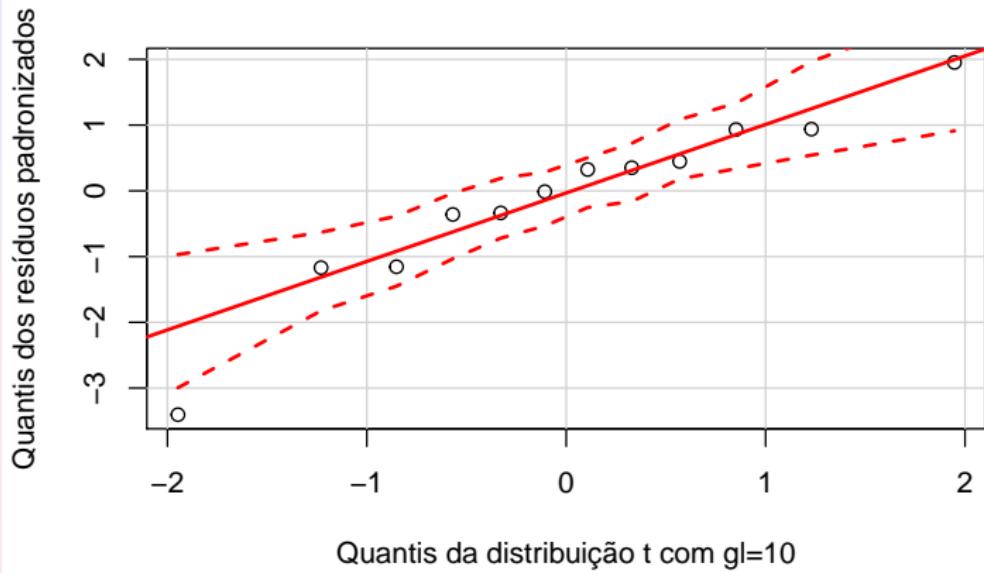


Figura 8: Gráfico QQ correspondente ajuste do modelo de regressão linear aos dados da Tabela 1 (com todas as observações).

## RLS-Gráficos QQ

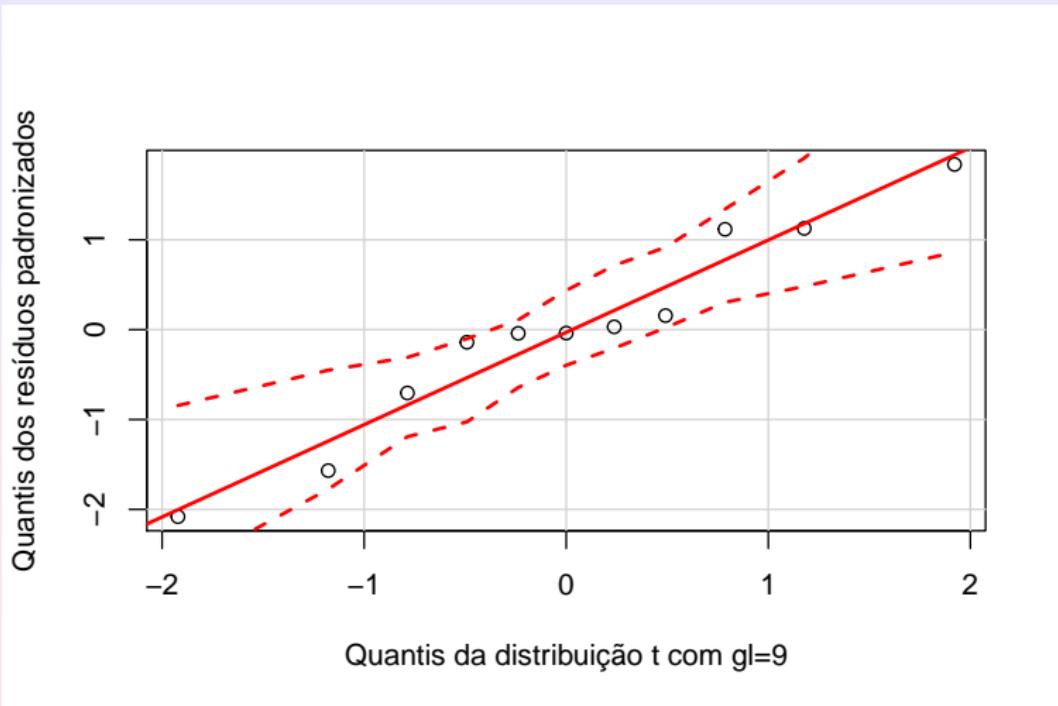


Figura 9: Gráfico QQ correspondente ajuste do modelo de regressão linear aos dados da Tabela 1 (sem a observação influente).

## RLS-Dados correlacionados

**Exemplo:** Na Tabela 2 apresentamos valores do peso de um bezerro observado a cada duas semanas após o nascimento com o objetivo de avaliar seu crescimento nesse período. O gráfico de dispersão correspondente está disposto na Figura 10.

Tabela 2: Peso (kg) de um bezerro nas primeiras 26 semanas após o nascimento

Semana	Peso
0	32,0
2	35,5
4	39,2
6	43,7
8	51,8
10	63,4
12	76,1
14	81,1
16	84,6
18	89,8
20	97,4
22	111,0
24	120,2
26	134,2

## RLS-Dados correlacionados

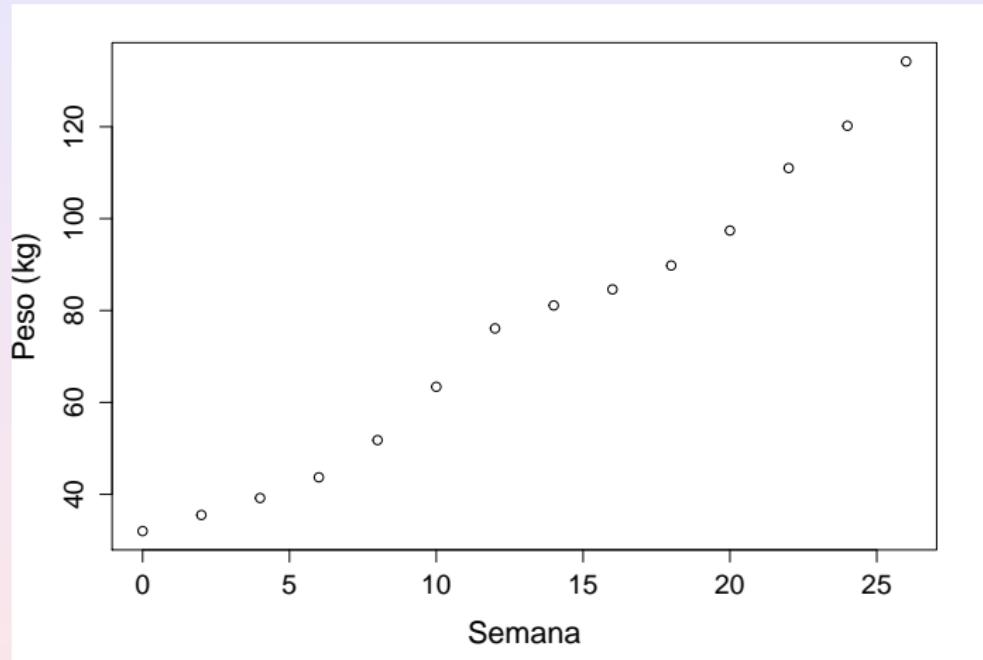


Figura 10: Gráfico de dispersão para os dados da Tabela 2

## Dados correlacionados

- Tendo em vista o gráfico de dispersão, um possível modelo seria

$$y_t = \alpha + \beta t + \gamma t^2 + e_t, \quad (5)$$

$i = 1, \dots, 14$  em que  $y_t$  representa o peso do bezerro no instante  $t$ ,  $\alpha$  denota o valor esperado de seu peso ao nascer,  $\beta$  e  $\gamma$  representam os componentes linear e quadrático da curva que rege a variação temporal do peso no intervalo de tempo estudado e  $e_t$  denota um erro aleatório.

Utilizamos  $t$  como índice para salientar que as observações são colhidas sequencialmente ao longo do tempo.

- O coeficiente de determinação ajustado,  $R_{aj}^2 = 0,987$  indica que o ajuste (por mínimos quadrados) do modelo com  $\hat{\alpha} = 29,9$  (2,6),  $\hat{\beta} = 2,7$  (2,5) e  $\hat{\gamma} = 0,05$  (0,02) é excelente (sob essa ótica, obviamente).
- Por outro lado, o gráfico de resíduos apresentado na Figura 11 mostra sequências de resíduos positivos seguidas de sequências de resíduos negativos, sugerindo uma possível correlação positiva entre eles (autocorrelação).

## RLS-Dados correlacionados

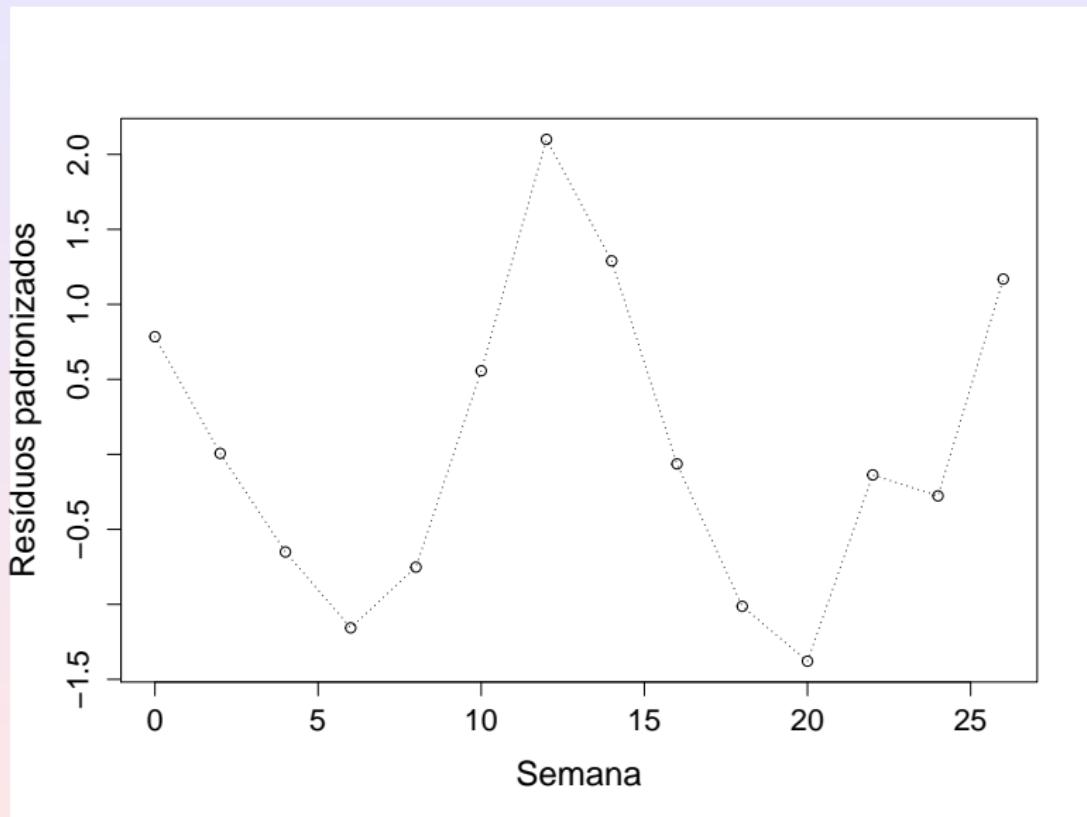


Figura 11: Resíduos studentizados obtidos do ajuste do modelo (5)

## RLS-Dados correlacionados

- Uma maneira de contornar esse problema, é modificar os componentes aleatórios do modelo para incorporar essa possível autocorrelação nos erros. Nesse contexto, podemos considerar o modelo (5) com

$$e_t = \rho e_{t-1} + u_t, \quad t = 1, \dots, n \quad (6)$$

em que  $u_t \sim N(0, \sigma^2)$ ,  $t = 1, \dots, n$ , independentes e  $e_0$  é uma constante (geralmente igual a zero). Essas suposições implicam que  $\text{Var}(e_t) = \sigma^2/(1 - \rho^2)$  e que  $\text{Cov}(e_t, e_{t-s}) = \rho^s[\sigma^2/(1 - \rho^2)]$ .

- Para testar a hipótese de que os erros são não correlacionados pode-se utilizar a **estatística de Durbin-Watson**:

$$D = \frac{\sum_{t=2}^n (\hat{e}_t - \hat{e}_{t-1})^2}{\sum_{t=1}^n \hat{e}_t^2}, \quad (7)$$

em que  $\hat{e}_t$ ,  $t = 1, \dots, n$  são os resíduos obtidos do ajuste do modelo (5) por mínimos quadrados.

## RLS-Dados correlacionados

- Expandindo (7) podemos verificar que

$$D \approx 2 - 2 \frac{\sum_{t=2}^n \hat{e}_t \hat{e}_{t-1}}{\sum_{t=1}^n \hat{e}_t^2}, \quad (8)$$

- Se os resíduos não forem correlacionados, então  $\sum_{t=2}^n \hat{e}_t \hat{e}_{t-1} \approx 0$  e consequentemente,  $D \approx 2$ ; se, por outro lado, os resíduos forem altamente correlacionados, esperamos que  $\sum_{t=2}^n \hat{e}_t \hat{e}_{t-1} \approx \sum_{t=2}^n \hat{e}_t^2$  e então  $D \approx 0$ ; finalmente, se os resíduos tiverem uma grande correlação negativa, esperamos que  $\sum_{t=2}^n \hat{e}_t \hat{e}_{t-1} \approx -\sum_{t=2}^n \hat{e}_t^2$  e nesse caso,  $D \approx 4$ .
- Durbin and Watson (1950), Durbin and Watson (1951) e Durbin and Watson (1971) produziram tabelas da distribuição da estatística  $D$  que podem ser utilizados para avaliar a suposição de que os erros são não correlacionados.
- O valor da estatística de Durbin-Watson para os dados do Exemplo sob o modelo (5) é  $D = 0,91$  ( $p < 0,0001$ ), sugerindo um alto grau de autocorrelação dos resíduos. Uma estimativa do coeficiente de autocorrelação  $\rho$  é 0,50. Nesse caso, o modelo (5) - (6) poderá ser ajustado pelo **método de mínimos quadrados generalizados** ou por métodos de **Séries Temporais**.

## RLS - Inferência

- a)  $E(\hat{\alpha}) = \alpha$  e  $E(\hat{\beta}) = \beta$ , ou seja, os EMQ são não enviesados.
- b)  $\text{var}(\hat{\alpha}) = \sigma^2 \sum_{i=1}^n x_i^2 / [n \sum_{i=1}^n (x_i - \bar{x})]^2$ .
- c)  $\text{var}(\hat{\beta}) = \sigma^2 / \sum_{i=1}^n (x_i - \bar{x})^2$ .
- d)  $\text{cov}(\hat{\alpha}, \hat{\beta}) = -\sigma^2 \bar{x} / \sum_{i=1}^n (x_i - \bar{x})^2$ .

# RLS - Inferência

Com a suposição adicional de normalidade, pode-se mostrar que

e)  $y_i \sim N(\alpha + \beta x_i, \sigma^2)$

f) as estatísticas

$$t_{\hat{\alpha}} = \frac{\hat{\alpha} - \alpha}{S} \sqrt{\frac{n \sum (x_i - \bar{x})^2}{\sum x_i^2}}$$

e

$$t_{\hat{\beta}} = \frac{\hat{\beta} - \beta}{S} \sqrt{\sum (x_i - \bar{x})^2}$$

têm distribuição  $t$  de Student com  $(n - 2)$  graus de liberdade. Nesse contexto, os resíduos padronizados também seguem uma distribuição  $t$  de Student com  $(n - 2)$  graus de liberdade. Daí a denominação alternativa de resíduos studentizados

- g) Com esses resultados é possível testar as hipóteses  $H_0 : \alpha = 0$  e  $H_0 : \beta = 0$ , bem como construir intervalos de confiança para esses parâmetros.

# RLS - Inferência

- **Teorema de Gauss-Markov:** EMQ têm variância mínima na classe dos estimadores não enviesados que sejam funções lineares das observações  $y_i$  (que não depende da suposição de normalidade dos erros).
- Quando os erros não seguem uma distribuição Normal, mas o tamanho da amostra é suficientemente grande, pode-se mostrar com o auxílio do **Teorema Limite Central** que sob certas condições de regularidade (usualmente satisfeitas na prática), os estimadores  $\hat{\alpha}$  e  $\hat{\beta}$  têm distribuições aproximadamente normais com variâncias que podem ser estimadas pelas expressões indicadas nos itens b) e c).

## RLS - Previsão

- Um dos objetivos da análise de regressão é fazer previsões sobre a variável resposta com base em valores das variáveis explicativas.
- Uma estimativa para o valor esperado  $E(Y|X = x_0)$  da variável resposta  $Y$  dado um valor  $x_0$  da variável explicativa é  $\hat{y} = \hat{\alpha} + \hat{\beta}x_0$  e com base nos resultados anteriores pode-se mostrar que a variância de  $\hat{y}$  é

$$\text{var}(\hat{y}) = \sigma^2 \left[ \frac{1}{n} + \frac{(x_0 - \bar{x})^2}{\sum_{i=1}^n (x_i - \bar{x})^2} \right].$$

- Então os limites superior e inferior para um intervalo de confiança aproximado com coeficiente de confiança de 95% para o valor esperado de  $Y$  dado  $X = x_0$  são

$$\hat{y} \pm 1,96 S \sqrt{\frac{1}{n} + \frac{(x_0 - \bar{x})^2}{\sum_{i=1}^n (x_i - \bar{x})^2}}$$

com  $S^2$  denotando uma estimativa de  $\sigma^2$ . Podemos dizer que esse intervalo deve conter o verdadeiro valor esperado de  $E(Y|X = x)$ , i.e., a média de  $Y$  para todas as observações em que  $X = x_0$ , com coeficiente de confiança de 95%.

## RLS - Previsão

Isso não significa que esperamos que o intervalo contenha o verdadeiro valor de  $Y$ , digamos  $Y_0$  para uma unidade de investigação para a qual  $X = x_0$ . Nesse caso precisamos levar em conta a variabilidade de  $Y|X = x_0$  em torno de seu valor esperado  $E(Y|X = x_0)$ .

Como  $Y_0 = \hat{y} + e_0$  sua variância é

$$\text{var}(Y_0) = \text{var}(\hat{y}) + \text{var}(e_0) = \sigma^2 \left[ \frac{1}{n} + \frac{(x_0 - \bar{x})^2}{\sum_{i=1}^n (x_i - \bar{x})^2} \right] + \sigma^2$$

Então os limites superior e inferior de um **intervalo de previsão** (aproximado) para  $Y_0$ , com  $\gamma = 95\%$ , são

$$\hat{y} \pm 1,96S \sqrt{1 + \frac{1}{n} + \frac{(x_0 - \bar{x})^2}{\sum_{i=1}^n (x_i - \bar{x})^2}}.$$

Note que se aumentarmos indefinidamente o tamanho da amostra, a amplitude do intervalo de confiança para o valor esperado tenderá para zero, porém a amplitude do intervalo de previsão correspondente a uma unidade específica tenderá para  $2 \times 1,96 \times \sigma$ .

# Referências

- Morettin, P. A. and Singer, J. M. (2022). *Estatística e Ciência de Dados*. LTC, Rio de Janeiro.
- James, G., Witten, D., Hastie, T. and Tibshirani, R. (2017). *Introduction to Statistical Learning*. Springer.

## Apêndice: Quantis empíricos

- Suponha que a v.a.  $X$  tenha distribuição contínua, com f.d.a.  $F$ . Então, para  $0 \leq p \leq 1$ , o  $p$ -quantil de  $F$  é o valor  $Q_p$  satisfazendo  $F(Q_p) = p$ , ou seja,

$$F(Q_p) = P(X \leq Q_p) = p.$$

Se existir a inversa de  $F$ , então  $Q_p = F^{-1}(p)$ . No caso de  $X$  ser discreta, a definição tem que ser modificada: o  $p$ -quantil é o valor  $Q_p$  satisfazendo

$$\begin{aligned} P(X \leq Q_p) &\geq p, \\ P(X \geq Q_p) &\geq 1 - p. \end{aligned}$$

- Dado um conjunto de observações, podemos calcular os *quantis empíricos*. Uma maneira é considerar a *função de distribuição empírica*  $\hat{F}_n$  como estimador de  $F$ , ou seja, dadas as observações  $X_1, \dots, X_n$  de  $X$ ,

$$\hat{F}_n(x) = \frac{1}{n} \#\{i : 1 \leq i \leq n, X_i \leq x\}.$$

Então, o quantil  $Q_p$  é estimado pelo  $p$ -quantil de  $\hat{F}_n$ . Ou seja, o  $p$ -quantil estimado,  $q_p$ , seria definido por  $\hat{F}_n(q_p) = p$ . Contudo, usaremos um enfoque um pouco diferente.

## Apêndice: Quantis empíricos

- Chamemos de  $X_1, \dots, X_T$  os valores observados e considere as estatísticas de ordem  $X_{(1)} \leq X_{(2)} \leq \dots \leq X_{(T)}$ . Um estimador consistente de  $Q_p$  é dado pelo  $p$ -quantil empírico, definido por

$$q_p = \begin{cases} X_{(i)}, & \text{se } p = p_i = (i - 0,5)/T, i = 1, \dots, T \\ (1 - f_i)X_{(i)} + f_i X_{(i+1)}, & \text{se } p_i < p < p_{i+1} \\ X_{(1)}, & \text{se } 0 < p < p_1 \\ X_{(T)}, & \text{se } p_T < p < 1, \end{cases}$$

onde  $f_i = (p - p_i)/(p_{i+1} - p_i)$ .

- Ou seja, ordenados os dados,  $q_p$  é uma das estatísticas de ordem, se  $p$  for da forma  $p_i = (i - 0,5)/T$  e está na reta ligando os pontos  $(p_i, X_{(i)})$  e  $(p_{i+1}, X_{(i+1)})$ , se  $p$  estiver entre  $p_i$  e  $p_{i+1}$ . Tomamos  $p_i$  da forma escolhida e não como  $i/T$  para que, por exemplo, a mediana calculada segundo esta definição coincida com a definição usual.

## Apêndice: Quantis empíricos

Há dois tipos de gráficos  $Q \times Q$ : teóricos e empíricos.

- O primeiro tipo é usado para verificar se um conjunto de dados vem de determinada distribuição.
- O segundo tipo é usado para verificar se dois conjuntos de dados têm uma mesma distribuição.
- Para verificar se um conjunto de dados provém de uma distribuição especificada, consideramos o gráfico em que, no eixo horizontal, colocamos os quantis teóricos da distribuição hipotetizada para os dados, e no eixo vertical, os quantis empíricos dos dados, ambos calculados nos pontos  $p_i$ , acima. Se as observações realmente são provenientes da distribuição em questão, os pontos deverão estar distribuídos ao longo de uma reta.

# MAE 5905: Introdução à Ciência de Dados

Pedro A. Morettin

Instituto de Matemática e Estatística  
Universidade de São Paulo  
pam@ime.usp.br  
<http://www.ime.usp.br/~pam>

## Aula 5

30 de março de 2023

# Sumário

## 1 Regressão Linear Múltipla

## 2 Regressão Logística

## RLM-modelo

- Com  $p$  variáveis explicativas  $X_1, \dots, X_p$  e uma variável resposta  $Y$ , o **modelo de regressão linear múltipla** é expresso como

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \dots + \beta_p x_{ip} + e_i, \quad i = 1, \dots, n. \quad (1)$$

O coeficiente  $\beta_0$  é o chamado **intercepto** e a variável explicativa associada a ele,  $x_{i0}$ , tem valor constante igual a 1. Para completar a especificação do modelo, supõe-se que os erros  $e_i$  são não correlacionados, tenham média zero e variância comum (desconhecida)  $\sigma^2$ .

- Se quisermos testar hipóteses a respeito dos coeficientes do modelo ou construir intervalos de confiança para eles por meio de estatísticas com distribuições exatas, a suposição de que a distribuição de frequências dos erros é Normal deve ser adicionada. O modelo (1) tem  $p + 2$  parâmetros desconhecidos, a saber,  $\beta_0, \beta_1, \dots, \beta_p$  e  $\sigma^2$ , que precisam ser estimados com base nos dados observados.

## RLM-modelo

- Definindo  $x_{i0} = 1$ ,  $i = 1, \dots, n$ , podemos escrever (1) na forma

$$y_i = \sum_{j=0}^p \beta_j x_{ij} + e_i, \quad i = 1, \dots, n.$$

Minimizando a soma dos quadrados do erros  $e_i$ , i.e.,

$$Q(\beta_0, \dots, \beta_p) = \sum_{i=1}^n e_i^2 = \sum_{i=1}^n [y_i - \sum_{j=0}^p \beta_j x_{ij}]^2,$$

em relação a  $\beta_0, \dots, \beta_p$  obtemos os **estimadores de mínimos quadrados**(EMQ)  $\hat{\beta}_j$ ,  $j = 1, \dots, p$ , de modo que

$$\hat{y}_i = \sum_{j=0}^p \hat{\beta}_j x_{ij}, \quad i = 1, \dots, n$$

são os **valores estimados** (sob o modelo).

- Os termos

$$\hat{e}_i = y_i - \hat{y}_i, \quad i = 1, \dots, n \tag{2}$$

são os **resíduos**, cuja análise é fundamental para avaliar se modelos da forma (1) se ajustam bem aos dados.

## RLM - o modelo

Para efeitos computacionais os dados correspondentes a problemas de regressão linear múltipla devem ser dispostos como indicado na Tabela 1.

Tabela 1: Matriz de dados

$Y$	$X_1$	$X_2$	$\cdots$	$X_p$
$y_1$	$x_{11}$	$x_{12}$	$\cdots$	$x_{1p}$
$y_2$	$x_{21}$	$x_{22}$	$\cdots$	$x_{2p}$
$\vdots$	$\vdots$	$\vdots$		$\vdots$
$y_n$	$x_{n1}$	$x_{n2}$	$\cdots$	$x_{np}$

Em geral, a variável correspondente ao intercepto (que é constante e igual a um) não precisa ser incluída na matriz de dados; os pacotes computacionais incluem-na naturalmente no modelo a não ser que se indique o contrário.

## RLM - o modelo

Para facilitar o desenvolvimento metodológico, convém expressar o modelo na forma matricial

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \mathbf{e}. \quad (3)$$

em que  $\mathbf{y} = (y_1, \dots, y_n)^\top$  é o vetor cujos elementos são os valores da variável resposta  $Y$ ,  $\mathbf{X} = (\mathbf{1}, \mathbf{x}_1, \dots, \mathbf{x}_p)$  é a matriz cujos elementos são os valores das variáveis explicativas, com  $\mathbf{x}_j = (x_{1j}, \dots, x_{nj})^\top$  contendo os valores da variável  $X_j$ ,  $\boldsymbol{\beta} = (\beta_0, \beta_1, \dots, \beta_p)^\top$  contém os respectivos coeficientes e  $\mathbf{e} = (e_1, \dots, e_n)^\top$  é o vetor de **erros aleatórios**.

## RLM - Exemplo

- Os dados **esteira** são provenientes de um estudo cujo objetivo é avaliar o efeito do índice de massa corpórea (IMC) e da carga aplicada numa esteira ergométrica no consumo de oxigênio (VO2) numa determinada fase do exercício.
- Para associar a distribuição do consumo de oxigênio ( $Y$ ) com as informações sobre carga na esteira ergométrica ( $X_1$ ) e IMC ( $X_2$ ), consideramos o seguinte modelo de regressão linear múltipla:

$$y_i = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + e_i, \quad (4)$$

$i = 1, \dots, 28$  com as suposições usuais sobre os erros (média zero, variância constante  $\sigma^2$  e não correlacionados). Aqui, o parâmetro  $\beta_1$  representa a variação no VO2 esperada por unidade carga para indivíduos com o mesmo IMC. O parâmetro  $\beta_2$  tem interpretação semelhante com a substituição de carga na esteira por IMC e IMC por carga na esteira.

## RLM - Exemplo

- Como não temos dados para indivíduos com IMC menor que 17,50 e carga menor que 32, o parâmetro  $\beta_0$  deve ser interpretado como um fator de ajuste do plano que aproxima a verdadeira função que relaciona o valor esperado da variável resposta com as variáveis explicativas na região em que há dados disponíveis.
- Se substituíssemos  $X_1$  por  $X_1 - 32$  e  $X_2$  por  $X_2 - 17.5$ , o termo  $\beta_0$  corresponderia ao VO2 esperado para um indivíduo com IMC = 17,50 submetido a uma carga igual a 32 na esteira ergométrica.

O modelo (4) pode ser expresso na forma matricial (3) com

$$\mathbf{y} = \begin{bmatrix} 14,1 \\ 16,3 \\ \vdots \\ 31,0 \end{bmatrix}, \quad \mathbf{X} = \begin{bmatrix} 1 & 24,32 & 71 \\ 1 & 27,68 & 91 \\ \vdots & \vdots & \vdots \\ 1 & 24,34 & 151 \end{bmatrix}, \quad \boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \end{bmatrix}, \quad \mathbf{e} = \begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_{28} \end{bmatrix}.$$

- Para problemas com diferentes tamanhos de amostra ( $n$ ) e diferentes números de variáveis explicativas ( $p$ ), basta alterar o número de elementos do vetor de respostas  $\mathbf{y}$  e do vetor de coeficientes  $\boldsymbol{\beta}$  e modificar a matriz com os valores das variáveis explicativas, alterando o número de linhas e colunas convenientemente.

## RLM - Propriedades

- Uma das vantagens da expressão do modelo de regressão linear múltipla em notação matricial é que o método de mínimos quadrados utilizado para estimar o vetor de parâmetros  $\beta$  no modelo (3) pode ser desenvolvido de maneira universal e corresponde à minimização da forma quadrática

$$Q(\beta) = \mathbf{e}^\top \mathbf{e} = (\mathbf{y} - \mathbf{X}\beta)^\top (\mathbf{y} - \mathbf{X}\beta) = \sum_{i=1}^n e_i^2. \quad (5)$$

- Por meio da utilização de operações matriciais, obtém-se a seguinte expressão para os estimadores de mínimos quadrados

$$\hat{\beta} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}. \quad (6)$$

- Sob a suposição de que  $E(\mathbf{e}) = \mathbf{0}$  e  $\text{var}(\mathbf{e}) = \sigma^2 \mathbf{I}_n$ , em que  $\mathbf{I}_n$  denota a matriz identidade de dimensão  $n$ , temos

- $E(\hat{\beta}) = \beta,$
- $\text{var}(\hat{\beta}) = \sigma^2 (\mathbf{X}^\top \mathbf{X})^{-1}.$

## RLM - Propriedades

- Além disso, se adicionarmos a suposição de que os erros têm distribuição Normal, pode-se mostrar que o estimador (6) tem uma distribuição Normal multivariada, o que permite a construção de intervalos de confiança para ou testes de hipóteses sobre os elementos (ou combinações lineares deles) de  $\beta$  por meio de estatísticas com distribuições exatas. Mesmo sem a suposição de normalidade para os erros, um recurso ao **Teorema Limite Central** permite mostrar que a distribuição aproximada do estimador (6) é Normal, com média a  $\beta$  e matriz de covariâncias  $\sigma^2(\mathbf{X}^\top \mathbf{X})^{-1}$ .
- Um estimador não enviesado de  $\sigma^2$  é

$$\begin{aligned}s^2 &= [n - (p + 1)]^{-1} (\mathbf{y} - \hat{\mathbf{X}}\hat{\beta})^\top (\mathbf{y} - \hat{\mathbf{X}}\hat{\beta}) \\ &= [n - (p + 1)]^{-1} \mathbf{y}^\top [\mathbf{I}_n - \mathbf{X}(\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top] \mathbf{y}.\end{aligned}$$

- Com duas variáveis explicativas, o gráfico de dispersão precisa ser construído num espaço tridimensional, que ainda pode ser representado em duas dimensões; para mais que 2 variáveis explicativas, o gráfico de dispersão requer um espaço com mais do que três dimensões que não pode ser representado no plano. Por isso, uma alternativa é construir gráficos de dispersão entre a variável resposta e cada uma das variáveis explicativas.

## RLM - Gráficos

Para os dados **esteira**, o gráfico de dispersão com três dimensões incluindo o plano correspondente ao modelo de regressão múltipla ajustado está disposto na Figura 1.

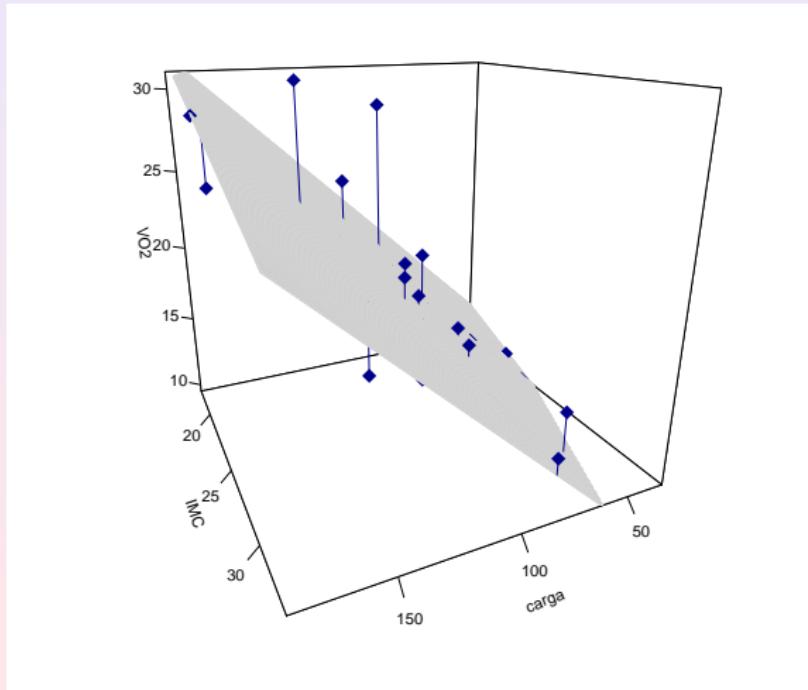


Figura 1: Gráfico de dispersão tridimensional para os dados esteira

## RLM - Gráficos

Os gráficos de dispersão correspondentes a cada uma das duas variáveis explicativas estão dispostos na Figura 2 e indicam que a distribuição do VO<sub>2</sub> varia positivamente com a carga na esteira e negativamente com o IMC.

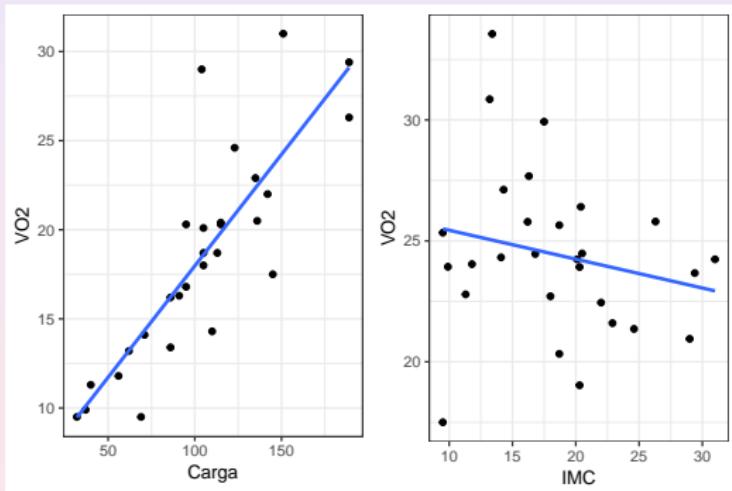


Figura 2: Gráficos de dispersão para os dados **esteira**.

## RLM - Uso do R

O uso da função lm() conduz aos seguintes resultados.

Call:

lm(formula = VO2 ~ IMC + carga, data = esteira)

Coefficients:

	Estimate	Std. Error	t value	Pr(>  t )
Intercept	15.44726	4.45431	3.468	0.00191 **
IMC	-0.41317	0.17177	-2.405	0.02389 *
carga	0.12617	0.01465	8.614	5.95e - 09 ***

Residual standard error: 3.057 on 25 degrees of freedom

Multiple R-squared: 0.759, Adjusted R-squared: 0.7397

F-statistic: 39.36 on 2 and 25 DF, p-value: 1.887e - 08

## RLM - Uso do R

- Essa saída nos diz que os coeficientes (erro padrão) correspondentes ao ajuste do modelo (4) aos dados **esteira** são  $\hat{\beta}_0 = 15,45$  (4,45),  $\hat{\beta}_1 = 0,13$  (0,01) e  $\hat{\beta}_2 = -0,41$  (0,17). Então, segundo o modelo, o valor esperado do VO2 para um indivíduo (IMC fixado) aumenta de 0,13 unidades para cada aumento de uma unidade da carga na esteira; similarmente, o valor esperado do VO2 para indivíduos submetidos à mesma carga na esteira diminui de 0,41 unidades com o aumento de uma unidade no IMC.
- Embora o coeficiente de determinação  $R^2 = 0,74$  sugira a adequação do modelo, convém avaliá-la por meio de outras ferramentas diagnósticas. No caso de regressão linear múltipla, gráficos de resíduos podem ter cada uma das variáveis explicativas ou os valores ajustados no eixo das abscissas. Para o exemplo, esses gráficos estão dispostos na Figura 3 juntamente com o gráfico contendo as distâncias de Cook.
- Os gráficos de resíduos padronizados não indicam um comprometimento da hipótese de homoscedasticidade embora seja possível suspeitar de dois ou três pontos discrepantes (correspondentes aos indivíduos com identificação 4, 8 e 28) que também são salientados no gráfico das distâncias de Cook. Veja também a Figura 1.

## RLM - Uso do R

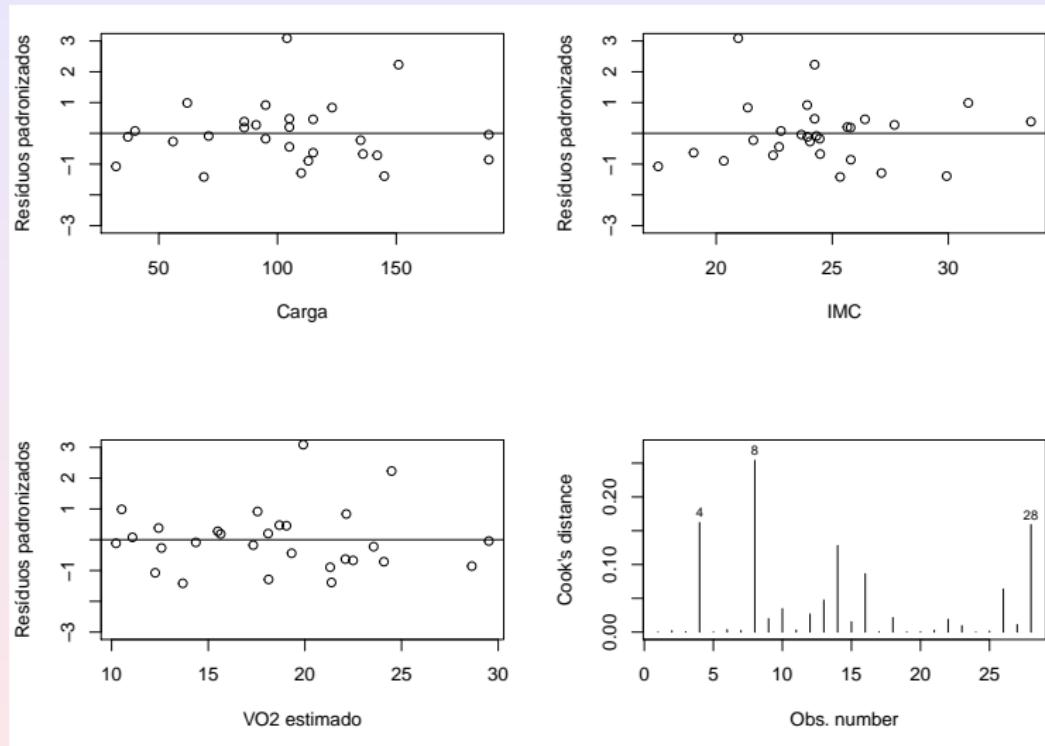


Figura 3: Gráficos de resíduos padronizados e distâncias de Cook para o ajuste do modelo (4) aos dados esteira

frame

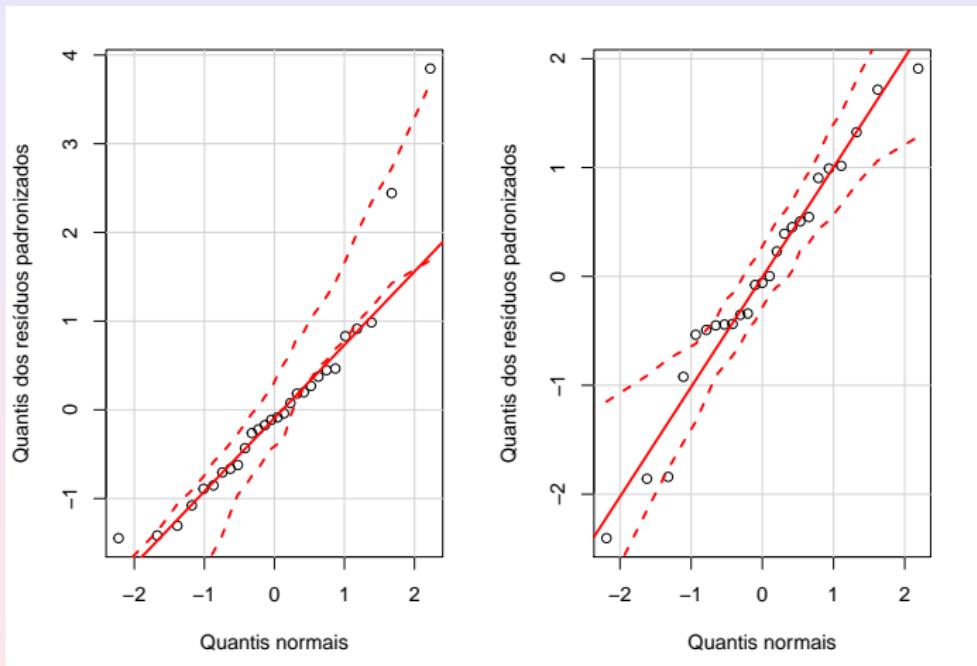


Figura 4: Gráficos QQ correspondentes ao ajuste do modelo (4) aos dados **esteira** com (painel esquerdo) e sem (painel direito) os pontos com identificação 4, 8 e 28.

## RL - Regressão logística

- **Exemplo.** O conjunto de dados **inibina** foi obtido de um estudo cuja finalidade era avaliar a utilização da inibina B como marcador da reserva ovariana de pacientes submetidas à fertilização *in vitro*. A variável explicativa é a diferença entre a concentração sérica de inibina B após estímulo com o hormônio FSH (hormônio folículo estimulante) e sua concentração sérica pré estímulo e a variável resposta é a classificação das pacientes como boas ou más respondedoras com base na quantidade de óócitos recuperados.
- A diferença entre esse problema e aqueles estudados nas seções anteriores está no fato de a variável resposta ser dicotômica e não contínua. Se definirmos a variável  $Y$  com valor igual a 1 no caso de resposta positiva e igual a zero no caso de resposta negativa, a resposta média será igual à proporção  $p = E(Y)$  de pacientes com resposta positiva. Essencialmente, o objetivo da análise é modelar essa proporção como função da variável explicativa.

## RL - o modelo

- Em vez de modelar a resposta média, convém modelar uma função dela, a saber o logaritmo da chance de resposta positiva para evitar estimativas de proporções com valores fora do intervalo  $(0, 1)$ . O modelo correspondente pode ser escrito como

$$\log \frac{P(Y_i = 1|X = x)}{P(Y_i = 0|X = x)} = \alpha + \beta x_i, \quad i = 1, \dots, n. \quad (7)$$

- De forma equivalente,

$$P(Y_i = 1|X = x) = \frac{\exp(\alpha + \beta x_i)}{1 + \exp(\alpha + \beta x_i)}, \quad i = 1, \dots, n. \quad (8)$$

## RL - o modelo

- Neste contexto, o parâmetro  $\alpha$  é interpretado como o logaritmo da chance de resposta positiva para pacientes com  $x_i = 0$  (concentrações de inibina pré e pós estímulo iguais) e o parâmetro  $\beta$  corresponde ao logaritmo da razão entre a chance de resposta positiva para pacientes com diferença de uma unidade na variável explicativa.
- O ajuste desse modelo é realizado pelo método de máxima verossimilhança. A função de verossimilhança a ser maximizada é

$$\ell(\alpha, \beta | \mathbf{x}, \mathbf{y}) = \prod_{i=1}^n [p(x_i)]^{y_i} [1 - p(x_i)]^{1-y_i}$$

$$p(x_i) = \frac{\exp(\alpha + \beta x_i)}{1 + \exp(\alpha + \beta x_i)}.$$

## RL - EMV

- A maximização da verossimilhança pode ser concretizada por meio da maximização de seu logaritmo

$$L(\alpha, \beta | \mathbf{x}, \mathbf{y}) = \sum_{i=1}^n \left\{ y_i \log[p(x_i)] + (1 - y_i) \log[1 - p(x_i)] \right\}.$$

- Os estimadores de máxima verossimilhança de  $\alpha$  e  $\beta$  correspondem à solução das **equações de estimação**

$$\sum_{i=1}^n \left\{ y_i - \frac{\exp(\hat{\alpha} + \hat{\beta}x_i)}{1 + \exp(\hat{\alpha} + \hat{\beta}x_i)} \right\} = 0 \quad \text{e} \quad \sum_{i=1}^n x_i \left\{ y_i - \frac{\exp(\hat{\alpha} + \hat{\beta}x_i)}{1 + \exp(\hat{\alpha} + \hat{\beta}x_i)} \right\} = 0.$$

- Como esse sistema de equações não tem solução explícita, deve-se recorrer a métodos iterativos como o **método de Newton-Raphson**.

## RL - Uso do R

O uso da função `glm()` produz os resultados a seguir:

Call:

```
glm(formula = resposta ~ difinib, family = binomial, data = dados)
```

Deviance Residuals:

Min	1Q	Median	3Q	Max
-1.9770	-0.5594	0.1890	0.5589	2.0631

Coefficients:

	Estimate	Std. Error	z value	Pr(> z )
Intercept	-2.310455	0.947438	-2.439	0.01474
inib	0.025965	0.008561	3.033	0.00242

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 43.230 on 31 degrees of freedom

Residual deviance: 24.758 on 30 degrees of freedom

AIC: 28.758

Number of Fisher Scoring iterations: 6

- As estimativas dos parâmetros (com erro padrão entre parênteses)  $\alpha$  e  $\beta$  correspondentes ao modelo ajustado aos dados **inibina** são, respectivamente,

$$\hat{\alpha} = -2,31 \ (0,95), \quad \hat{\beta} = 0,03 \ (0,01)$$

- Consequentemente, a chance de resposta positiva para pacientes com mesmo nível de inibina B pré e pós estímulo hormonal é  $\exp(\hat{\alpha}) = 0,10$ .
- Essa chance fica multiplicada por  $\exp(\hat{\beta}) = 1,03$  para cada aumento de uma unidade na diferença entre os níveis de inibina B pré e pós estímulo hormonal.
- Os erros padrões de  $\exp(\hat{\alpha})$  e  $\exp(\hat{\beta})$  são calculados por meio do **método Delta**. Ver Nota de Capítulo 6.

## RL - Uso do R

A função predict() pode ser usada para estimar a probabilidade de que a resposta seja positiva, dados os valores da variável explicativa. Algumas dessas probabilidades estão indicadas abaixo:

1	2	3	4	5	6
0.1190483	0.7018691	0.9554275	0.9988353	0.5797138	0.9588247
7	8	9	10		
0.8045906	0.8362005	0.9534173	0.8997726		

Por exemplo, o valor 0,1190483 foi obtido calculando-se

$$P(Y = 1|X = 11, 90) = \frac{\exp\{-2,310455 + (0,025965)(11,90)\}}{1 + \exp\{-2,310455 + (0,025965)(11,90)\}}. \quad (9)$$

## RL - uso do R

- Para prever se a resposta vai ser positiva ou negativa, temos que converter essas probabilidades previstas em rótulos de classes, “positiva” / ou “negativa”. Considerando respostas positivas como aquelas cuja probabilidade seja maior do que 0,7, digamos, podemos utilizar a função `table()` para obter a seguinte tabela:

		resposta	
		pred	negativa positiva
pred	negativa	11	5
	positiva	2	14

- Os elementos da diagonal dessa tabela indicam os números de observações corretamente classificadas. Ou seja, a proporção de respostas corretas será  $(11+14)/32 = 78\%$ . Esse valor depende do limiar fixado, 0,7, no caso. Um *default* usualmente fixado é 0,5, e nesse caso, a proporção de respostas corretas vai aumentar.
- A utilização de Regressão Logística nesse contexto de classificação será detalhada no Capítulo 10.

## Algumas considerações

- O modelo de RLM tem dois aspectos importantes: aditividade e linearidade.
- **aditividade** significa que o efeito de mudanças em um preditor  $X_j$  sobre a resposta  $Y$  é independente dos valores dos demais preditores.
- **linearidade** significa que uma mudança em  $Y$  devida a uma mudança unitária em  $X_j$  é constante, independentemente do valor de  $X_j$ .
- Uma maneira de estender o modelo linear é incluir **interações**, por exemplo,

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_1 X_2 + e.$$

- Outra maneira: considerar **regressão polinomial**. Nesse caso, temos uma função não linear, mas o modelo continua linear!
- Para verificar se há necessidade de um modelo não linear, fazer o gráfico dos resíduos *versus*  $x_i$ , no caso de RLS e de resíduos *versus*  $\hat{y}_i$ , no de RLM.

## Referências

Morettin, P. A. and Singer, J. M. (2022). *Estatística e Ciência de Dados*. LTC: Rio de Janeiro.

James, G., Witten, D., Hastie, T. and Tibshirani, R. (2017). *Introduction to Statistical Learning*. Springer.

# MAE 5905: Introdução à Ciência de Dados

Pedro A. Morettin

Instituto de Matemática e Estatística  
Universidade de São Paulo  
[pam@ime.usp.br](mailto:pam@ime.usp.br)  
<http://www.ime.usp.br/~pam>

## Aula 6

17 de abril de 2023

# Sumário

## 1 Regularização

- Regularização Ridge
- Regularização Lasso
- Outras propostas

## 2 Regularização: teoria

## 3 Modelos aditivos generalizados

## Um exemplo

- Consideremos um exemplo [proposto em Bishop (2006)] cujo objetivo é ajustar um modelo de regressão polinomial a um conjunto de 10 pontos gerados por meio da expressão  $y_i = \text{sen}(2\pi x_i) + e_i$  em que  $e_i$  segue um distribuição Normal com média nula e variância  $\sigma^2$ .
- Os dados estão representados na Figura 1 por pontos em azul. A curva verde corresponde a  $y_i = \text{sen}(2\pi x_i)$ ; em vermelho estão representados os ajustes baseados em regressões polinomiais de graus, 0, 1, 3 e 9.
- Claramente, a curva baseada no polinômio do terceiro grau consegue reproduzir o padrão da curva geradora dos dados sem, no entanto, predizer os dados com total precisão. A curva baseada no polinômio de grau 9, por outro lado, tem um ajuste perfeito, mas não reproduz o padrão da curva utilizada para gerar os dados. Esse fenômeno é conhecido como **sobreajuste**.

## Um exemplo

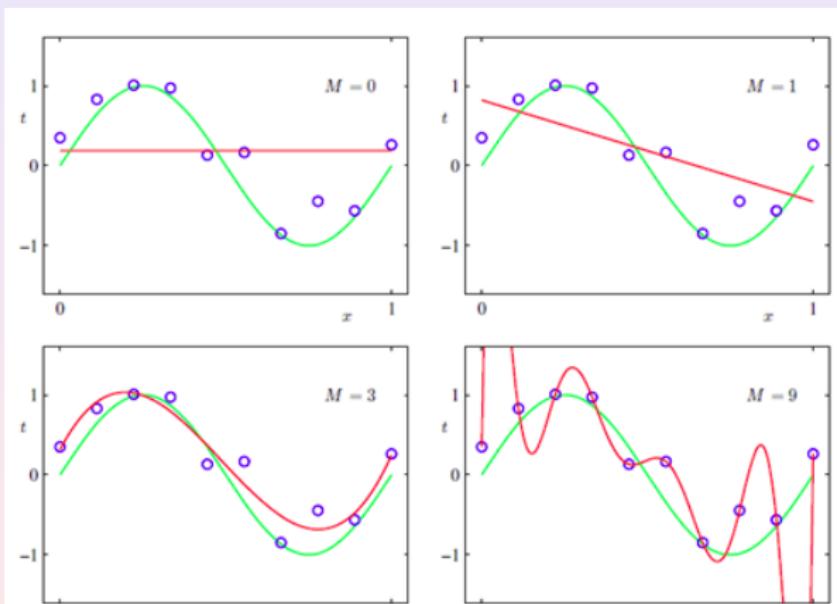


Figura 1: Ajuste de modelos polinomiais a um conjunto de dados hipotéticos.

## Quando usar EMQ?

- Se a relação entre resposta e preditores for aproximadamente linear, então um modelo de regressão linear múltipla (RLM) pode ser adequado e estimadores de mínimos quadrados (EMQ) tenderão a ter baixo viés e conduzir a previsões boas.
- Se  $n \gg p$ , EMQ terão também baixa variância.
- Se  $n$  não for muito maior do que  $p$ , EMQ apresentarão muita variabilidade (sobreajuste) e previsões ruins.
- Se  $p > n$ , não existirão EMQ univocamente determinados.  
Aumentando-se o número de variáveis,  $R^2 \rightarrow 1$ , o EQM de treinamento tenderá zero e o EQM de teste crescerá. **Não use MQ!**

## Possíveis abordagens

Possíveis alternativas para remover variáveis irrelevantes de um modelo de RLM, de modo a obter maior interpretabilidade:

- **Seleção de subconjuntos de variáveis** (**subset selection**) ; vários procedimentos podem ser usados (stepwise (forward e backward), uso de critérios de informação (AIC, BIC),  $R^2$  ajustado, validação cruzada).
- **Encolhimento (shrinkage)**: usa todos os preditores mas os coeficientes são encolhidos para zero; pode funcionar para selecionar variáveis. Reduz variância. Também chamada **regularização**.
- **Redução da dimensão**: projetar os preditores sobre um subespaço de dimensão menor  $q < p$ , que consiste em obter combinações lineares (ou projeções) dos preditores. Essas  $q$  projeções são usadas como novos preditores no ajuste de MQ, ACP, AF, ICA.

# Regularização

- O termo **regularização** refere-se a um conjunto de técnicas utilizadas para especificar modelos que se ajustem a um conjunto de dados evitando o **sobreajuste (overfitting)**.
- Essencialmente, essas técnicas servem para ajustar modelos de regressão em que a função de perda contém um termo de penalização cuja finalidade é reduzir a influência de coeficientes responsáveis por flutuações excessivas.
- Embora haja várias técnicas de regularização, consideraremos apenas a regularização  $L_2$ , ou **Ridge**, a regularização  $L_1$  ou **Lasso** (*least absolute shrinkage and selection operator*) e uma mistura dessas duas, chamada de **Elastic net**.

# Regularização

- O termo de regularização da técnica Lasso usa uma soma de valores absolutos dos parâmetros e um **coeficiente de penalização** que os encolhe para zero. Essa técnica serve para seleção de modelos, porque associa pesos nulos a parâmetros não significativos.
- Isso implica uma **solução esparsa** (Dizemos que um modelo é esparso se a maioria dos elementos do correspondente vetor de parâmetros é nula ou desprezável).
- Na regularização  $L_2$ , por outro lado, o termo de regularização usa uma soma de quadrados dos parâmetros e um coeficiente de penalização que força alguns pesos a serem pequenos, mas não os anula e consequentemente não conduz a soluções esparsas. Essa técnica de regularização não é robusta com relação a valores atípicos, pois pode conduzir a valores muito grandes do termo de penalização.

## O contexto

- Consideraremos o modelo de regressão

$$y_i = \beta_0 + \beta_1 x_{1i} + \dots + \beta_p x_{pi} + e_i, \quad i = 1, 2, \dots, n, \quad (1)$$

ou

$$y_i = \beta_0 + \beta_i^\top \mathbf{x}_i + e_i, \quad (2)$$

com as  $p$  variáveis preditoras reunidas no vetor  $\mathbf{x}_i = (x_{1i}, \dots, x_{pi})^\top$ ,  $y_i$  representando a variável resposta,  $e_i$  indicando as inovações de média zero e  $\boldsymbol{\beta} = (\beta_0, \beta_1, \dots, \beta_p)^\top$  denotando o vetor de parâmetros a serem estimados.

- Vamos considerar  $\mathbf{x}_i = (\mathbf{x}_i(1)^\top, \mathbf{x}_i(2)^\top)^\top$ , com  $\mathbf{x}_i(1) \in \mathbb{R}^s$  o vetor de variáveis **relevantes** e  $\mathbf{x}_i(2) \in \mathbb{R}^{p-s}$  o vetor de variáveis **irrelevantes**,  $\boldsymbol{\beta} = (\boldsymbol{\beta}(1)^\top, \boldsymbol{\beta}(2)^\top)^\top$ .
- Objetivos:**
  - Selecione o conjunto de variáveis correto:  $\hat{\boldsymbol{\beta}}(1) \neq 0$  e  $\hat{\boldsymbol{\beta}}(2) = 0$  ([seleção do modelo](#));
  - Estime  $\boldsymbol{\beta}(1)$  como se o conjunto de variáveis correto fosse conhecido.

# Regularização Ridge

- Supomos adicionalmente que  $\beta_0 = 0$  e consideremos estimadores de mínimos quadrados (EMQ) penalizados da forma

$$\hat{\beta}_{\text{Ridge}}(\lambda) = \arg \min_{\beta} \left[ \frac{1}{2n} \sum_{t=1}^n (y_t - \beta^\top \mathbf{x}_t)^2 + \lambda \sum_{j=1}^p \beta_j^2 \right], \quad (3)$$

em que  $\lambda$  é o coeficiente de regularização, que controla o número de parâmetros do modelo. Se  $\lambda = \infty$ , não há variáveis a serem incluídas no modelo e se  $\lambda = 0$ , obtemos os EMQ usuais.

- Dizemos que  $\hat{\beta}_{\text{Ridge}}(\lambda)$  é o **estimador Ridge**.

# Ridge - Propriedades

Pode-se mostrar que

$$\hat{\beta}_{\text{Ridge}}(\lambda) = (\mathbf{X}^\top \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^\top \mathbf{y}, \quad (4)$$

em que  $\mathbf{X} = (\mathbf{x}_1^\top, \dots, \mathbf{x}_n^\top)^\top$  é a matriz de especificação do modelo e  $\mathbf{y} = (y_1, \dots, y_n)^\top$ .

Alguns resultados sobre as propriedades dessa classe de estimadores são:

- 1) Em geral, o estimador *Ridge* não é consistente. Sua consistência assintótica vale quando  $\lambda = v\lambda_n \rightarrow \infty$ ,  $\lambda_n/n \rightarrow 0$  e  $p < n$ .
- 2) O estimador *Ridge* é enviesado para os parâmetros não nulos.
- 3) A técnica de regularização *Ridge* não serve para a seleção de modelos.
- 4) A escolha do coeficiente de regularização  $\lambda$  pode ser feita via validação cruzada ou por meio de algum critério de informação.
- 5) A técnica de regressão *Ridge* foi introduzida por Hoerl e Kennard (1970) para tratar do problema da multicolinearidade.

## Ridge - Propriedades

Obter o mínimo em (3) é equivalente a minimizar a soma de quadrados não regularizada sujeita à restrição

$$\sum_{j=1}^p \beta_j^2 \leq m, \quad (5)$$

para algum valor apropriado  $m$ , ou seja, é um problema de otimização com multiplicadores de Lagrange.

Na Figura 2 (a) apresentamos um esquema com o valor ótimo do vetor  $\beta$ , a região circular correspondente à restrição (5) e os círculos representando as curvas de nível da função erro não regularizada.

## Ridge - Propriedades

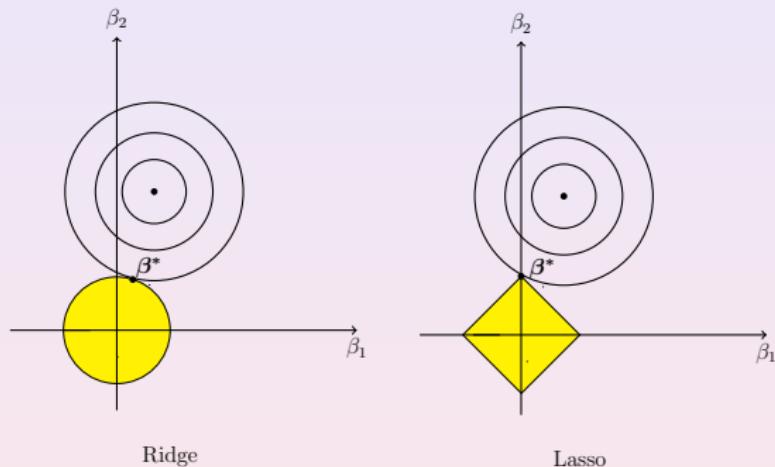


Figura 2: Esparsidade do modelo: (a) Ridge; (b) Lasso.

## Regularização Lasso

- Consideremos, agora, o **estimador Lasso**, obtido de

$$\widehat{\boldsymbol{\beta}}_{\text{Lasso}}(\lambda) = \arg \min_{\boldsymbol{\beta}} \left[ \frac{1}{2n} \sum_{t=1}^n (y_t - \boldsymbol{\beta}^\top \mathbf{x}_t)^2 + \lambda \sum_{j=1}^p |\beta_j| \right], \quad (6)$$

- Neste caso, a restrição (5) é substituída por

$$\sum_{j=1}^p |\beta_j| \leq m, \quad (7)$$

- No painel (b) da Figura 2 (b), podemos observar que a regularização Lasso pode gerar uma solução esparsa, ou seja com  $\beta_1^* = 0$ .

## Regularização Lasso

- Existe uma correspondência 1 – 1 entre as formulações (6) e (7): para cada valor de  $m$  para a qual (7) vale, existe um valor de  $\lambda$  que fornece a mesma solução para (6). Reciprocamente, a solução  $\hat{\beta}(\lambda)$  de (6) resolve o problema restrito, com  $m = \|\hat{\beta}(\lambda)\|_1$ .
- Tanto no caso Ridge, como no Lasso, a constante  $1/2n$  pode ser substituída por  $1/2$  ou mesmo  $1$ . Essa padronização torna os valores de  $\lambda$  comparáveis para diferentes tamanhos amostrais (por exemplo ao usar CV).
- Em análise convexa, a condição necessária e suficiente para a solução de (6) é

$$-\frac{1}{n} \langle \mathbf{x}_j, \mathbf{y} - \mathbf{X}\beta \rangle + \lambda s_j = 0, \quad j = 1, \dots, p, \quad (8)$$

onde  $s_j$  é uma quantidade desconhecida, igual a  $\text{sinal}(\beta_j)$ , se  $\beta_j \neq 0$  e algum valor no intervalo  $[-1, 1]$ , se  $\beta_j = 0$  (sub-gradiente para a função valor absoluto).

- O sistema (8) é uma forma das chamadas condições de Karush-Kuhn-Tucker (KKT) para o problema (6).

## Lasso - Propriedades

Algumas propriedades estatísticas do estimador Lasso:

- 1) O estimador Lasso encolhe para zero os parâmetros que correspondem a preditores redundantes.
- 2) O estimador é enviesado para parâmetros não nulos.
- 3) Sob certas condições, o estimador Lasso seleciona as variáveis relevantes do modelo atribuindo pesos nulos aos respectivos coeficientes.
- 4) O estimador não é consistente em geral.
- 5) Quando  $p = n$ , ou seja, quando o número de variáveis preditoras é igual ao número de observações, a técnica Lasso corresponde à aplicação de um **limiar suave** (*soft threshold*) a  $Z_j = \mathbf{x}_j^\top \mathbf{y} / n$ , ou seja,

$$\hat{\beta}_j(\lambda) = \text{sinal}(Z_j) (|Z_j| - \lambda/2)_+, \quad (9)$$

em que  $(x)_+ = \max\{x, 0\}$ .

## Threshold dado por (9)

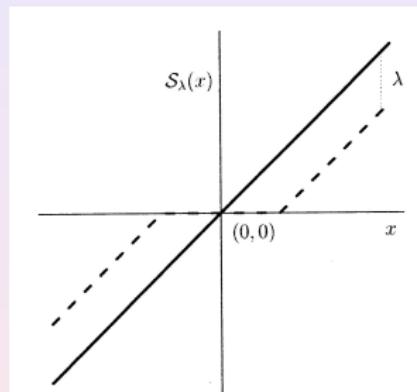


Figura 3: Threshold: MQ (linha cheia) e Lasso (linha tracejada), para o caso  $p = n$ .

## Elastic net

- O estimador *Elastic net* é

$$\widehat{\boldsymbol{\beta}}_{\text{EN}}(\lambda_1, \lambda_2) = \arg \min_{\boldsymbol{\beta}} \sum_{t=1}^n \frac{1}{2n} (y_t - \boldsymbol{\beta}^\top \mathbf{x}_t)^2 + \lambda_2 \sum_{i=1}^p \beta_i^2 + \lambda_1 \sum_{i=1}^p |\beta_i|. \quad (10)$$

- Na Figura 4 apresentamos esquematicamente uma região delimitada pela restrição  $J(\boldsymbol{\beta}) \leq m$ , em que  $J(\boldsymbol{\beta}) = \alpha \sum_{j=1}^p \beta_j^2 + (1 - \alpha) \sum_{j=1}^p |\beta_j|$ , para algum  $m$ , com  $\alpha = \lambda_2 / (\lambda_1 + \lambda_2)$ , além daquelas delimitadas pelas restrições *Ridge* e *Lasso*.
- Pode-se mostrar que sob determinadas condições, o estimador *Elastic Net* é consistente.

## Elastic net

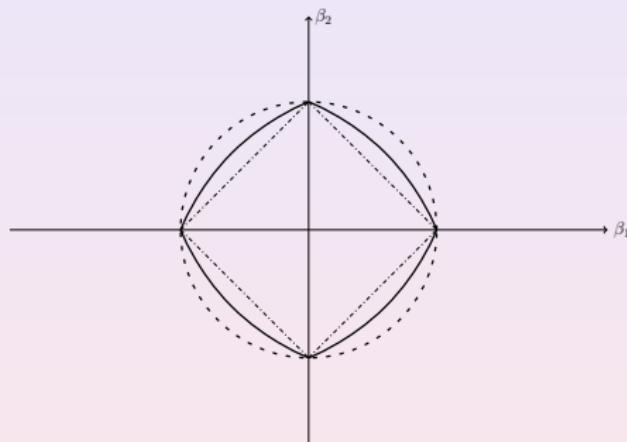


Figura 4: Geometria das restrições *Elastic Net* (curva contínua), *Ridge* (curva tracejada) e *Lasso* (curva pontilhada).

## Lasso adaptativo

- O estimador **Lasso adaptativo** (adaLASSO) é dado por

$$\hat{\beta}_{AL}(\lambda) = \arg \min_{\beta} \frac{1}{2n} \sum_{t=1}^n (y_t - \beta^\top \mathbf{x}_t)^2 + \lambda \sum_{i=1}^p w_i |\beta_i|, \quad (11)$$

em que  $w_1, \dots, w_p$  são pesos não negativos pré-definidos.

- Usualmente, toma-se  $w_j = |\tilde{\beta}_j|^{-\tau}$ , para  $0 < \tau \leq 1$  e  $\tilde{\beta}_j$  é um estimador inicial (por exemplo o estimador Lasso).
- O estimador **Lasso adaptativo** é consistente sob condições não muito fortes.
- A função `adalasso` do pacote `parcor` do R pode ser usada para calcular esse estimador.
- O pacote `glmnet` do R pode ser usado para obter estimadores Lasso e *Elastic net* sob modelos de regressão linear, regressão logística e multinomial, regressão Poisson além de modelos de Cox. Para detalhes, veja Friedman et al. (2010).

## Comparação entre os métodos

- Tanto Ridge como o Lasso encolhem os coeficiente para zero. No caso do Lasso, a penalidade  $L_1$  tem a finalidade de tornar alguns dos coeficientes serem efetivamente nulos. Logo, o Lasso realiza **seleção de modelos**.
- Lasso resulta em modelos **esparsos**, ou seja, mais fáceis de interpretar.
- Tanto no Ridge, quanto no Lasso, a variância decresce e o viés cresce à medida que  $\lambda$  cresce.
- Ridge tem desempenho melhor que o Lasso nos caso que um número grande de preditores tem relação com a variável resposta. Em caso contrário, o Lasso tem desempenho melhor (em termos de EQM).
- Em geral, nenhum método domina os outros em **todas** as situações.

## Viés da regularização Ridge

Supondo  $p < n$ , fazendo  $\mathbf{R} = \mathbf{X}^\top \mathbf{X}$ , e usando a expressão do estimador *ridge* (3), obtemos

$$\begin{aligned}\hat{\beta}_{\text{Ridge}}(\lambda) &= (\mathbf{X}^\top \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^\top \mathbf{y} \\ &= (\mathbf{R} + \lambda \mathbf{I})^{-1} \mathbf{R} [\mathbf{R}^{-1} \mathbf{X}^\top \mathbf{y}] \\ &= [\mathbf{R}(\mathbf{I} + \lambda \mathbf{R}^{-1})]^{-1} \mathbf{R} ((\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}) \\ &= (\mathbf{I} + \lambda \mathbf{R}^{-1})^{-1} \mathbf{R}^{-1} \mathbf{R} \hat{\beta}_{\text{MQ}} \\ &= (\mathbf{I} + \lambda \mathbf{R}^{-1}) \hat{\beta}_{\text{MQ}},\end{aligned}$$

em que  $\hat{\beta}_{\text{MQ}}$  denota o estimador de mínimos quadrados ordinários. Tomando a esperança condicional da expressão anterior, dada  $\mathbf{X}$ , obtemos

$$\begin{aligned}E[\hat{\beta}_{\text{Ridge}}(\lambda) | \mathbf{X}] &= E[(\mathbf{I} + \lambda \mathbf{R}^{-1}) \hat{\beta}_{\text{MQ}}] \\ &= (\mathbf{I} + \lambda \mathbf{R}^{-1}) \beta,\end{aligned}$$

de onde segue

$$E[\hat{\beta}_{\text{Ridge}}(\lambda)] = (\mathbf{I} + \lambda \mathbf{R}^{-1}) \beta \neq \beta.$$

## Ridge: um resultado

Pode-se provar que

$$\hat{\beta}_{\text{Ridge}}(\lambda) = \mathbf{V} \text{diag} \left( \frac{d_1}{d_1^2 + \lambda}, \frac{d_2}{d_2^2 + \lambda}, \dots, \frac{d_p}{d_p^2 + \lambda} \right) \mathbf{U}^\top \mathbf{y},$$

em que  $\mathbf{X} = \mathbf{UDV}^\top$  é a decomposição em valores singulares de  $\mathbf{X}$ , com  $\mathbf{U}$  denotando uma matriz ortogonal de dimensão  $n \times p$ ,  $\mathbf{V}$  uma matriz ortogonal de dimensão  $p \times p$  e  $\mathbf{D}$  uma matriz diagonal com dimensão  $p \times p$ , contendo os correspondentes valores singulares  $d_1 \geq d_2 \geq \dots \geq d_p \geq 0$  (raízes quadradas dos autovalores de  $\mathbf{X}^\top \mathbf{X}$ ).

## Ridge quando $\mathbf{X}$ é ortogonal

Quando  $\mathbf{X}^\top \mathbf{X} = \mathbf{I}$ , pode-se provar que:

1. Ridge e EMQ:

$$\hat{\beta}_{\text{Ridge}}(\lambda) = \frac{1}{1 + \lambda} \hat{\beta}_{\text{MQ}}. \quad (12)$$

2. A escolha ótima de  $\lambda$  minimizando o erro de previsão esperado é

$$\lambda^* = \frac{p\sigma^2}{\sum_{i=1}^p \beta_j^2}. \quad (13)$$

## Ridge e Lasso: escolha do parâmetro $\lambda$

- A escolha do parâmetro de regularização  $\lambda$  pode ser baseada em **validação cruzada** ou em algum **critério de informação**.
- Seja  $\Lambda = \{\lambda_1, \dots, \lambda_M\}$  uma grade de valores para  $\lambda$ . No segundo caso,

$$\hat{\lambda} = \arg \min_{\lambda \in \Lambda} [-\log \text{verossimilhança} + \text{penalização}],$$

como

$$AIC = \log[\hat{\sigma}^2(\lambda)] + gl(\lambda) \frac{2}{n},$$

$$BIC = \log[\hat{\sigma}^2(\lambda)] + gl(\lambda) \frac{\log n}{n},$$

$$HQ = \log[\hat{\sigma}^2(\lambda)] + gl(\lambda) \frac{\log \log n}{n},$$

em que  $gl(\lambda)$  é o número de graus de liberdade associado a  $\lambda$ , nomeadamente

$$gl(\lambda) = \text{tr} \left[ \mathbf{X} (\mathbf{X}^\top \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^\top \right] = \sum_{j=1}^p \frac{d_j^2}{d_j^2 + \lambda},$$

e

$$\hat{\sigma}^2(\lambda) = \frac{1}{n - gl(\lambda)} \sum_{i=1}^n [y_i - \hat{\beta}_{\text{Ridge}}(\lambda)^\top \mathbf{x}_i]^2.$$

## Ridge e Lasso: escolha do parâmetro $\lambda$

No caso de validação cruzada (VC):

- Calcule o erro da validação cruzada, como descrito abaixo, para cada valor de  $\lambda$  nessa grade.
- Escolha  $\lambda$  para o qual o erro da VC seja mínimo.
- O modelo é re-ajustado usando **todas** as observações disponíveis e o valor selecionado de  $\lambda$ .
- Pode-se usar o método LOOCV ou KFCV.

## Ridge - Consistência

- Quando  $\lambda = \lambda_n$  e  $p < n$ .
- O estimador Ridge pode ser escrito na forma

$$\begin{aligned}\hat{\beta}_{\text{Ridge}}(\lambda_n) &= (\mathbf{X}^\top \mathbf{X} + \lambda_n \mathbf{I})^{-1} \mathbf{X}^\top \mathbf{y} \\ &= \boldsymbol{\beta} - \lambda_n (n^{-1} \mathbf{X}^\top \mathbf{X} + \lambda_n \mathbf{I})^{-1} \boldsymbol{\beta} \\ &\quad + (n^{-1} \mathbf{X}^\top \mathbf{X} + \lambda_n \mathbf{I})^{-1} n^{-1} \mathbf{X} \mathbf{e}.\end{aligned}$$

- Quando  $\lambda_n \rightarrow 0$ , para  $n \rightarrow \infty$ , pelo Lema de Slutsky,

$$\begin{aligned}\lambda_n (n^{-1} \mathbf{X}^\top \mathbf{X} + \lambda_n \mathbf{I})^{-1} \boldsymbol{\beta} &\rightarrow 0, \\ (n^{-1} \mathbf{X}^\top \mathbf{X} + \lambda_n \mathbf{I})^{-1} n^{-1} \mathbf{X} \mathbf{e} &\rightarrow 0.\end{aligned}$$

- Pode ainda ser provado que, quando  $\sqrt{n} \lambda_n \rightarrow \lambda_0 \geq 0$ , quando  $n \rightarrow \infty$ ,

$$\sqrt{n}(\hat{\beta}_{\text{Ridge}} - \boldsymbol{\beta}) + \lambda_0 \mathbf{Q}^{-1} \boldsymbol{\beta} \rightarrow N(\mathbf{0}, \mathbf{Q}^{-1} \mathbf{V} \mathbf{Q}^{-1}),$$

onde  $\mathbf{Q} = p \lim_{n \rightarrow \infty} n^{-1} \mathbf{X}^\top \mathbf{X}$  e  $\mathbf{V}$  é a matriz de variância assintótica de  $n^{-1/2} \mathbf{X} \mathbf{e}$ .

## Lasso: teoria

Considere o caso de  $\mathbf{X}$  ortogonal e  $p < n$ . Então, pode-se provar que

$$\begin{aligned}\hat{\beta}_{\text{Lasso}}(\lambda) &= \arg \min_{\beta \in \mathbb{R}^p} \frac{1}{2} \|\mathbf{y} - \mathbf{X}\beta\|^2 + \lambda \|\beta\|_1 \\ &= \arg \min_{\beta \in \mathbb{R}^p} \left( -\mathbf{y}^\top \mathbf{X}\beta + \frac{1}{2} \|\beta\|_2^2 \right) + \lambda \|\beta\|_1 \\ &= \arg \min_{\beta \in \mathbb{R}^p} \sum_{i=1}^p \left( -\hat{\beta}_{MQ,i} \beta_i + \frac{1}{2} \beta_i^2 + \lambda |\beta_i| \right).\end{aligned}$$

- Como o problema é um programa de otimização quadrática com restrição convexa, a função objetivo torna-se uma soma de funções objetivos.
- Para cada  $i$ , minimiza-se  $Q_i = -\hat{\beta}_{MQ,i} \beta_i + \frac{1}{2} \beta_i^2 + \lambda |\beta_i|$ .

## Lasso: teoria

- No modelo RLM, suponha  $\mathbf{X}$  fixa ou iid e o erro normal, com média  $\mathbf{0}$  e variância  $\sigma^2 \mathbf{I}$ .
- O número de variáveis  $p = p_n$  e  $p >> n$ .
- No caso de  $\mathbf{X}$  fixa, resultados de consistência dependem da condição

$$\|\beta^0\|_1 = \|\beta_n^0\|_1 = o\left(\sqrt{\frac{n}{\log p}}\right).$$

- No caso de  $\mathbf{X}$  aleatória, sob condições sobre o erro e

$$\|\beta^0\|_1 = o\left(\left(\frac{n}{\log n}\right)^{1/4}\right),$$

e para  $\lambda$  da ordem de  $\sqrt{\log p/n}$ , o estimador Lasso é consistente:

$$[\hat{\beta}_{\text{Lasso}}(\lambda) - \beta^0] \Sigma [\hat{\beta}_{\text{Lasso}}(\lambda) - \beta^0]^T = o_p(1), \quad n \rightarrow \infty,$$

com  $\Sigma = \frac{1}{n} \mathbf{X}^\top \mathbf{X}$  no caso fixo ou  $\Sigma$  sendo a matriz de covariância de  $\mathbf{X}$ , no caso aleatório.

## Exemplo

Consideremos os dados do arquivo `antracose`, extraídos de um estudo cuja finalidade era avaliar o efeito da idade (`idade`), tempo vivendo em São Paulo (`tmunic`), horas diárias em trânsito (`htransp`), carga tabágica (`cargatabag`), classificação sócio-econômica (`ses`), densidade de tráfego na região onde habitou (`densid`) e distância mínima entre a residência a vias com alta intensidade de tráfego (`distmin`) num índice de antracose (`antracose`) que é uma medida de fuligem (*black carbon*) depositada no pulmão. Como esse índice varia entre 0 e 1, consideramos

$$\logrc = \log[\text{índice de antracose}/(1 - \text{índice de antracose})]$$

## Exemplo - MQ

Os estimadores de mínimos quadrados para um modelo linear podem ser obtidos por meio dos comandos

```
pulmao_lm <- lm(logrc ~ idade + tmunic + htransp + cargatabag +  
ses + densid + distmin, data=pulmao)
```

```
summary(pulmao_lm)
```

Coefficients:

	Estimate	Std. Error	t value	Pr(> t )
(Intercept)	-3.977e+00	2.459e-01	-16.169	< 2e-16 ***
idade	2.554e-02	2.979e-03	8.574	< 2e-16 ***
tmunic	2.436e-04	2.191e-03	0.111	0.911485
htransp	7.505e-02	1.634e-02	4.592	5.35e-06 ***
cargatabag	6.464e-03	1.055e-03	6.128	1.61e-09 ***
ses	-4.120e-01	1.238e-01	-3.329	0.000926 ***
densid	7.570e+00	6.349e+00	1.192	0.233582
distmin	3.014e-05	2.396e-04	0.126	0.899950

Residual standard error: 1.014 on 598 degrees of freedom

Multiple R-squared: 0.1965, Adjusted R-squared: 0.1871

F-statistic: 20.89 on 7 and 598 DF, p-value: < 2.2e-16

## Exemplo - Ridge

O ajuste dos modelos de regressão *Ridge*, *Lasso* ou *Elastic net* pode ser obtido com o pacote `glmnet`.

Utilizando esse pacote, ajustamos o modelo de regressão *Ridge* por meio de validação cruzada e obtivemos o gráfico da Figura 2 em que o erro quadrático médio (*MSE*) é expresso em função do logaritmo do coeficiente de regularização  $\lambda$ .

```
regridgecv = cv.glmnet(X, y, alpha = 0)
plot(regridgecv)
```

## Exemplo - Ridge

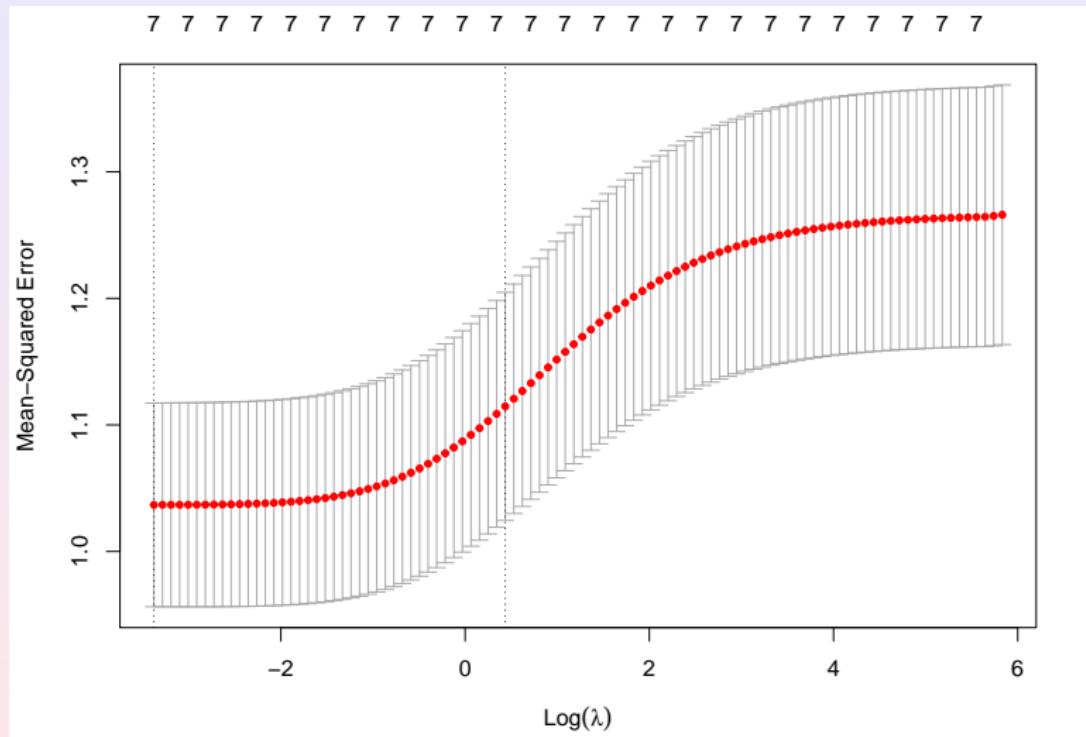


Figure 2: Gráfico para avaliação do efeito do coeficiente de regularização (Ridge).

## Exemplo - Ridge

Os coeficientes do ajuste correspondentes ao valor mínimo do coeficiente  $\lambda$ , juntamente com esse valor, são obtidos com os comandos

```
coef(regridgecv, s = "lambda.min")
8 x 1 sparse Matrix of class "dgCMatrix"
           1
(Intercept) -3.905299e+00
idade        2.456715e-02
tmunic       4.905597e-04
htransp       7.251095e-02
cargatabag   6.265919e-03
ses          -3.953787e-01
densid        7.368120e+00
distmin      3.401372e-05
> regridgecv$lambda.min
[1] 0.03410028
```

## Exemplo -Ridge

Com exceção das estimativas dos coeficientes das variáveis `tmunic` e `distmin` as demais foram encolhidas em direção a zero relativamente àquelas obtidas por mínimos quadrados. Os valores preditos e a correspondente raiz quadrada do *MSE* (usualmente denotada *RMSE*) são obtidos por meio de

```
predict(regridgecv, X, s = "lambda.min")
sqrt(regridgecv$cvm[regridgecv$lambda == regridgecv$lambda.min])
[1] 1.050218
```

## Exemplo - Lasso

O ajuste do modelo de regressão **Lasso** juntamente com o gráfico para a escolha do coeficiente  $\lambda$ , disposto na Figura 3, são obtidos com

```
reglassocv = cv.glmnet(X, y, alpha = 1)
plot(reglassocv)
```

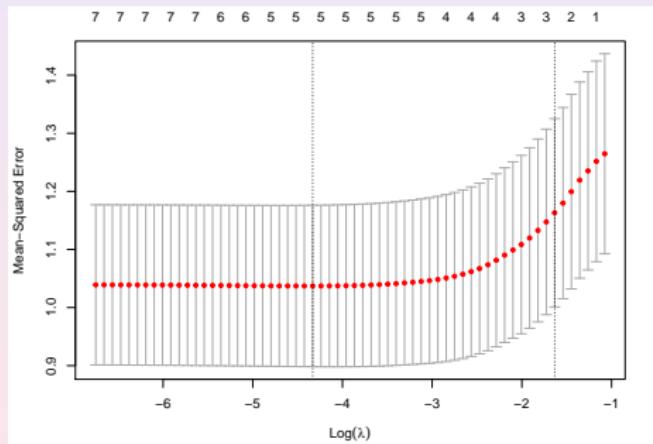


Figura 3: Gráfico para avaliação do efeito do coeficiente de regularização (*Lasso*).

## Exemplo - Lasso

Os coeficientes correspondentes à regularização *Lasso*, o valor mínimo do coeficiente  $\lambda$  e o *RMSE* são gerados por intermédio dos comandos

```
coef(reglassocv, s = "lambda.min")
8 x 1 sparse Matrix of class "dgCMatrix"
           1
(Intercept) -3.820975473
idade         0.024549358
tmunic        .
htransp       0.069750435
cargatabag   0.006177662
ses            -0.365713282
densid        5.166969594
distmin       .
reglassocv$lambda.min
[1] 0.01314064
sqrt(reglassocv$cvm[reglassocv$lambda == reglassocv$lambda.min])
[1] 1.018408
```

Neste caso, todos os coeficientes foram encolhidos em direção ao zero, e aqueles correspondentes às variáveis *tmunic* e *distmin* foram anulados.

## Exemplo - Elastic Net

Para o modelo **Elastic Net** com  $\alpha = 0,5$  os resultados são

```
regelncv = cv.glmnet(X, y, alpha = 0.5)
regelncv$lambda.min
[1] 0.02884367
coef(regelncv, s = "lambda.min")
8 x 1 sparse Matrix of class "dgCMatrix"
           1
(Intercept) -3.776354935
idade        0.024089256
tmunic       .
htransp      0.068289153
cargatabag   0.006070319
ses          -0.354080190
densid       4.889074555
distmin      .
sqrt(regelncv$cvm[regelncv$lambda == regelncv$lambda.min])
[1] 1.0183
```

Vemos que os 3 procedimentos resultam em EQM similares, com pequena vantagem para Elastic Net.

- Modelos lineares têm um papel muito importante na análise de dados, tanto pela facilidade de ajuste quanto de interpretação. De uma forma geral, os modelos lineares podem ser expressos como

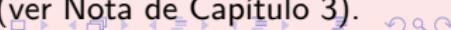
$$y_i = \beta_0 + \beta_1 f_1(x_{i1}) + \dots + \beta_p f_p(x_{ip}) + e_i \quad (14)$$

$i = 1, \dots, n$  em que as funções  $f_i$  são conhecidas. No modelo de regressão polinomial de segundo grau, por exemplo,  $f_1(x_{i1}) = x_{i1}$  e  $f_2(x_{i2}) = x_{i2}^2$ . Em casos mais gerais, poderíamos ter  $f_1(x_{i1}) = x_{ij}$  e  $f_2(x_{i2}) = \exp(x_{i2})$ . Em muitos problemas reais, no entanto, nem sempre é fácil especificar a forma das funções  $f_i$  e uma alternativa proposta por Hastie e Tibshirani (1996) são os chamados **Modelos Aditivos Generalizados** (*Generalized Additive Models* - GAM) que são expressos como (14) sem a especificação da forma das funções  $f_i$ .

- Quando a distribuição da variável resposta  $y_i$  pertence à **família exponencial**, o modelo pode ser considerado como uma extensão dos **Modelos Lineares Generalizados** (*Generalized Linear Models* - GLM) e é expresso como

$$g(\mu_i) = \beta_0 + \beta_1 f_1(x_{i1}) + \dots + \beta_p f_p(x_{ip}) \quad (15)$$

em que  $g$  é uma **função de ligação** e  $\mu_i = E(y_i)$  (ver Nota de Capítulo 3).



- Existem diversas propostas para a representação das funções  $f_i$  que incluem o uso de **splines naturais**, **splines suavizadas** e **regressões locais**.
- A suavidade dessas funções é controlada por parâmetros de suavização, que devem ser determinados *a priori*. Curvas muito suaves podem ser muito restritivas, enquanto curvas muito rugosas podem causar sobreajuste.
- O procedimento de ajuste dos modelos aditivos generalizados depende da forma escolhida para as funções  $f_i$ . A utilização de **splines naturais**, por exemplo, permite a aplicação direta do método de mínimos quadrados, graças à sua construção a partir de **funções base**.
- Para **splines penalizadas**, o processo de estimativa envolve algoritmos um pouco mais complexos, como aqueles conhecidos sob a denominação de **retroajustamento** (*backfitting*). Para detalhes sobre o ajuste dos modelos aditivos generalizados, consulte Hastie e Tibshirani (1990) e Hastie et al. (2008).

## Splines

- Para entender o conceito de *splines*, consideremos o seguinte modelo linear com apenas uma variável explicativa

$$y_i = \beta_0 + \beta_1 x_i + e_i, \quad i = 1, \dots, n. \quad (16)$$

- A ideia subjacente aos modelos aditivos generalizados é a utilização de funções base e consiste na substituição do termo  $\beta_1 x_i$  em (16) por um conjunto de transformações conhecidas  $b_1(x_i), \dots, b_t(x_i)$ , gerando o modelo

$$y_i = \alpha_0 + \alpha_1 b_1(x_i) + \dots + \alpha_t b_t(x_i) + e_i, \quad i = 1, \dots, n. \quad (17)$$

O modelo de regressão polinomial de grau  $t$  é um caso particular de (17) com  $b_j(x_i) = x_i^j$ ,  $j = 1, \dots, t$ .

## Splines

- Uma proposta para aumentar a flexibilidade da curva ajustada consiste em segmentar o domínio da variável preditora e ajustar diferentes polinômios de grau  $d$  aos dados de cada um dos intervalos gerados pela segmentação. Cada ponto de segmentação é chamado de **nó** e uma segmentação com  $k$  nós gera  $k + 1$  polinômios. Na Figura 43, apresentamos um exemplo com polinômios de terceiro grau e 4 nós.
- Nesse exemplo, a expressão (17) tem a forma

$$y_i = \begin{cases} \alpha_{01} + \alpha_{11}x_i + \alpha_{21}x_i^2 + \alpha_{31}x_i^3 + e_i, & \text{se } x_i \leq -0.5, \\ \alpha_{02} + \alpha_{12}x_i + \alpha_{22}x_i^2 + \alpha_{32}x_i^3 + e_i, & \text{se } -0.5 < x_i \leq 0, \\ \alpha_{02} + \alpha_{13}x_i + \alpha_{23}x_i^2 + \alpha_{33}x_i^3 + e_i, & \text{se } 0 < x_i \leq 0.5, \\ \alpha_{02} + \alpha_{14}x_i + \alpha_{24}x_i^2 + \alpha_{34}x_i^3 + e_i, & \text{se } 0.5 < x_i \leq 1, \\ \alpha_{05} + \alpha_{15}x_i + \alpha_{25}x_i^2 + \alpha_{35}x_i^3 + e_i, & \text{se } x_i > 1, \end{cases} \quad (18)$$

sendo que nesse caso, as funções base  $b_1(X), b_2(X), \dots, b_k(X)$  são construídas com a ajuda de funções indicadoras. Esse modelo é conhecido como **modelo polinomial cúbico segmentado**.

# Splines

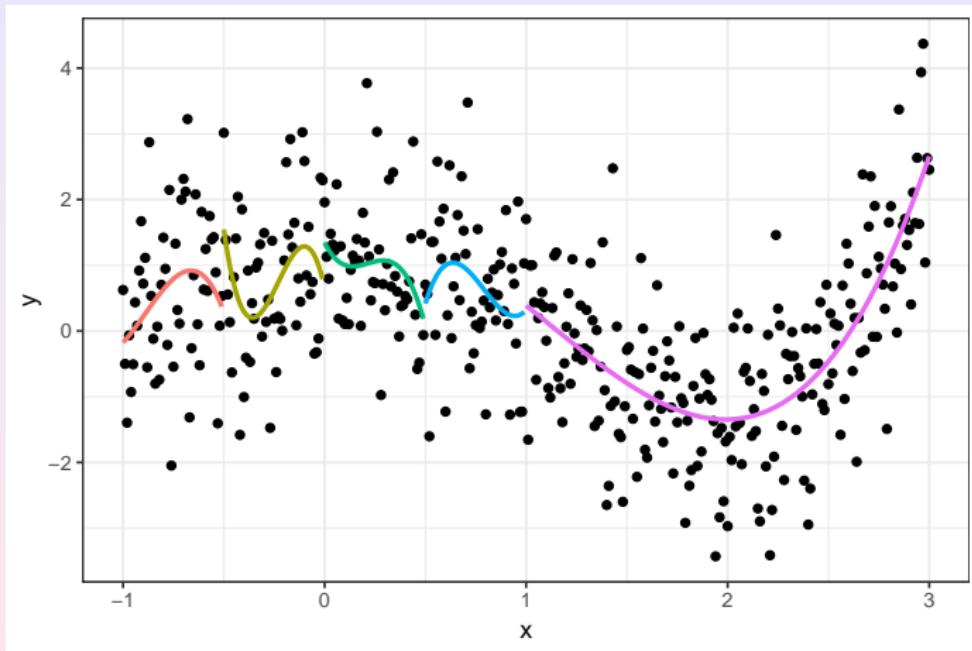


Figura 4: Polinômios de terceiro grau ajustados aos dados de cada região segmentada da variável  $X$ . Os nós são os pontos  $x = -0.5$ ,  $x = 0$ ,  $x = 0.5$  e  $x = 1$ .

## Splines

- A curva formada pela junção de cada um dos polinômios na Figura 43 não é contínua, apresentando “saltos” nos nós.  
Essa característica não é desejável, pois essas descontinuidades não são interpretáveis. Para contornar esse problema, podemos definir um *spline* de grau  $d$  como um polinômio segmentado de grau  $d$  com as  $d - 1$  primeiras derivadas contínuas em cada nó. Essa restrição garante a continuidade e suavidade (ausência de vértices) da curva obtida.
- Utilizando a representação por bases (17), um *spline* cúbico com  $k$  nós pode ser expresso como

$$y_i = \alpha_0 + \alpha_1 b_1(x_i) + \alpha_2 b_2(x_i) + \dots + \alpha_{k+3} b_{k+3}(x_i) + e_i, \quad i = 1, \dots, n, \quad (19)$$

com as funções  $b_1(x), b_2(x), \dots, b_{k+3}(x)$  escolhidas apropriadamente.

- Usualmente, essas funções envolvem três termos polinomiais, a saber,  $x$ ,  $x^2$  e  $x^3$  e  $k$  termos  $h(x, c_1), \dots, h(x, c_k)$  da forma

$$h(x, c_j) = (x - c_j)_+^3 = \begin{cases} (x - c_j)^3, & \text{se } x < c_j, \\ 0, & \text{em caso contrário,} \end{cases}$$

com  $c_1, \dots, c_k$  indicando os nós.

# Splines

- Com a inclusão do termo  $\alpha_0$ , o ajuste de um *spline* cúbico com  $k$  nós envolve a estimativa de  $k + 4$  parâmetros e, portanto, utiliza  $k + 4$  graus de liberdade. Mais detalhes sobre a construção desses modelos podem ser encontrados em Hastie (2008) e James et al. (2017).
- Além das restrições sobre as derivadas, podemos adicionar **restrições de fronteira**, exigindo que a função seja linear na região de  $x$  abaixo do menor nó e acima do maior nó. Essas restrições diminuem a variância dos valores extremos gerados pelo preditor, produzindo estimativas mais estáveis. Um *spline* cúbico com restrições de fronteira é chamado de **spline natural**.
- No ajuste de *splines* cúbicos ou naturais, o número de nós determina o grau de suavidade da curva e a sua escolha pode ser feita por validação cruzada conforme indicado em James et al. (2017). De uma forma geral, a maior parte dos nós é posicionada nas regiões do preditor com mais informação, isto é, com mais observações. Por pragmatismo, para modelos com mais de uma variável explicativa, costuma-se adotar o mesmo número de nós para todos os preditores.

# Splines

- Os *splines* suavizados constituem uma classe de funções suavizadoras que não utilizam a abordagem por funções bases. De maneira resumida, um *spline* suavizado é uma função  $f$  que minimiza

$$\sum_{i=1}^n [y_i - f(x_i)]^2 + \lambda \int f''(u)^2 du \quad (20)$$

em que  $f''$  corresponde à segunda derivada da função  $f$  e indica sua curvatura; quanto maior for a curvatura maior a penalização.

- O primeiro termo dessa expressão garante que  $f$  se ajustará bem aos dados, enquanto o segundo penaliza a sua variabilidade, isto é, controla a suavidade de  $f$ , que é regulada pelo parâmetro  $\lambda$ . A função  $f$  se torna mais suave conforme  $\lambda$  aumenta. A escolha desse parâmetro é geralmente feita por validação cruzada.
- Outro método bastante utilizado no ajuste funções não lineares para a relação entre a variável preditora  $X$  e a variável resposta  $Y$  é conhecido como **regressão local**. Esse método consiste em ajustar modelos de regressão simples em regiões em torno de cada observação  $x_0$  da variável preditora  $X$ .

# Splines

- Essas regiões são formadas pelos  $k$  pontos mais próximos de  $x_0$ , sendo que o parâmetro  $s = k/n$  determina o quanto suave ou rugosa será a curva ajustada. O ajuste é feito por meio de mínimos quadrados ponderados, com pesos inversamente proporcionais à distância entre cada ponto da região centrada em  $x_0$  e  $x_0$ . Aos pontos dessas regiões mais afastados de  $x_0$  são atribuídos pesos menores.
- **Lowess**: ajusta retas. Veja a Nota de Capítulo 5.2.
- Para uma excelente exposição sobre *splines* e penalização o leitor pode consultar Eilers e Marx (1996) e Eilers e Marx (2021).
- Modelos aditivos generalizados podem ser ajustados utilizando-se a função `gam()` do pacote `mgcv`. Essa função permite a utilização de *splines* como funções suavizadoras. Para regressão local, é necessário usar a função `gam()` do pacote `gam`. Também é possível utilizar o pacote `caret`, a partir da função `train()` e `method = "gam"`.

## MAG: exemplo

Consideremos os dados do arquivo `esforco` com o objetivo de prever os valores da variável `vo2fcrico` (VO2/FC no pico do exercício) a partir das variáveis `NYHA`, `idade`, `altura`, `peso`, `fcrep` (frequência cardíaca em repouso) e `vo2rep` (VO2 em repouso). Um modelo inicial de regressão linear múltipla também pode ser ajustado por meio dos seguintes comandos da função `gam`

```
mod0 <- gam(vo2fcrico ~ NYHA + idade + altura + peso + fcrep  
+ vo2rep, data=esforco)
```

Como não especificamos nem a distribuição da resposta, nem a função de ligação, a função `gam` utiliza a distribuição normal com função de ligação logarítmica, conforme indica o resultado.

## MAG: exemplo

```
Family: gaussian
Link function: identity
Formula:
vo2fcrico ~ NYHA + idade + altura + peso + fcrep + vo2rep
Parametric coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) -4.80229   4.43061  -1.084  0.280642  
NYHA1        -0.45757   0.50032  -0.915  0.362303  
NYHA2        -1.78625   0.52629  -3.394  0.000941 *** 
NYHA3        -2.64609   0.56128  -4.714  6.75e-06 *** 
NYHA4        -2.43352   0.70532  -3.450  0.000780 *** 
idade         -0.05670   0.01515  -3.742  0.000284 *** 
altura        0.09794   0.02654   3.690  0.000342 *** 
peso          0.08614   0.01739   4.953  2.48e-06 *** 
fcrep         -0.07096   0.01318  -5.382  3.84e-07 *** 
vo2rep        0.35564   0.24606   1.445  0.151033  
---
R-sq.(adj) =  0.607  Deviance explained = 63.5%
 GCV =  4.075  Scale est. = 3.7542    n = 127
```

## MAG: exemplo

Para avaliar a qualidade do ajuste, produzimos gráficos de dispersão entre os resíduos do ajuste e cada uma das variáveis preditoras. Esses gráficos estão dispostos na Figura 5 e sugerem relações possivelmente não lineares pelo menos em alguns casos.

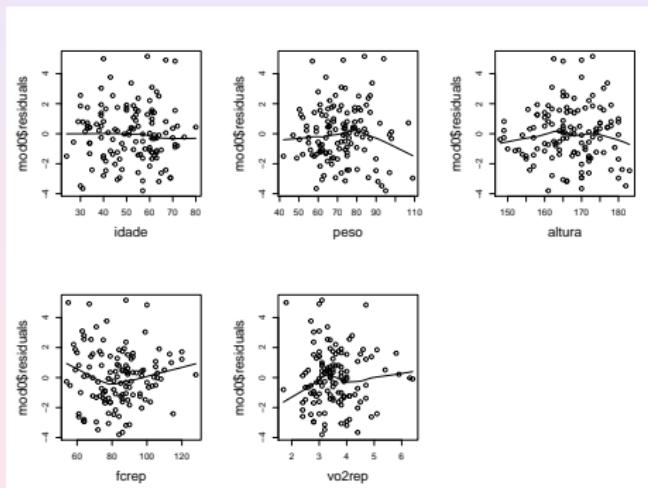


Figura 5: Gráficos de dispersão (com curva *lowess*) entre vo2fcpico e cada variável preditora contínua considerada no Exemplo.

## MAG: exemplo

Uma alternativa é considerar modelos GAM do tipo (14) em que as funções  $f_i$  são expressas em termos de *splines*. Em particular, um modelo GAM com *splines* cúbicos para todas as variáveis explicativas contínuas pode ser ajustado por meio do comando

```
mod1 <- gam(vo2fcrico ~ NYHA + s(idade) + s(altura) + s(peso) +  
           s(fcrep) + s(vo2rep), data=esforco)
```

gerando os seguintes resultados:

## MAG: exemplo

```
Family: gaussian
Link function: identity
Formula:
vo2fcrico ~ NYHA + s(idade) + s(altura) + s(peso) + s(fcrep) +
  s(vo2rep)
Parametric coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) 10.2101    0.3207 31.841 < 2e-16 ***
NYHA1       -0.5498    0.4987 -1.103  0.272614
NYHA2       -1.8513    0.5181 -3.573  0.000522 ***
NYHA3       -2.8420    0.5664 -5.018  1.99e-06 ***
NYHA4       -2.5616    0.7031 -3.643  0.000410 ***
---
Approximate significance of smooth terms:
            edf Ref.df   F p-value
s(idade) 1.000 1.000 15.860 0.00012 ***
s(altura) 5.362 6.476  3.751 0.00142 **
s(peso)   1.000 1.000 22.364 6.32e-06 ***
s(fcrep)  1.742 2.185 16.236 3.95e-07 ***
s(vo2rep) 1.344 1.615  0.906 0.47319
---
R-sq.(adj) =  0.64  Deviance explained = 68.2%
GCV = 3.9107  Scale est. = 3.435      n = 127
```

## MAG: exemplo

- O painel superior contém estimativas dos componentes paramétricos do modelo e o painel inferior, os resultados referentes aos termos suavizados. Neste caso apenas a variável categorizada NYHA não foi suavizada, dada sua natureza não paramétrica.
- A coluna rotulada edf contém os graus de liberdade efetivos associados a cada variável preditora. Para cada variável preditora contínua não suavizada, perde-se um grau de liberdade; para as variáveis suavizadas a atribuição de graus de liberdade é mais complexa em virtude do número de funções base e do número de nós utilizados no processo de suavização. A linha rotulada GCV (*Generalized Cross Validation*) está associada com a escolha (por validação cruzada) do parâmetro de suavização.
- A suavização é irrelevante apenas para a variável vo2rep e dado que ela também não apresentou contribuição significativa no modelo de regressão linear múltipla, pode-se considerar um novo modelo GAM obtido com a sua eliminação. Os resultados correspondentes, apresentados a seguir, sugerem que todas as variáveis preditoras contribuem significativamente para explicar sua relação com a variável resposta,

## MAG: exemplo

Formula:

vo2fcrico ~ NYHA + s(idade) + s(altura) + s(peso) + s(fcprep)

Parametric coefficients:

	Estimate	Std. Error	t value	Pr(> t )	
(Intercept)	10.2301	0.3202	31.948	< 2e-16	***
NYHA1	-0.5818	0.4985	-1.167	0.245650	
NYHA2	-1.8385	0.5161	-3.563	0.000539	***
NYHA3	-2.9669	0.5512	-5.382	4.04e-07	***
NYHA4	-2.4823	0.6980	-3.556	0.000551	***
---					

Approximate significance of smooth terms:

	edf	Ref.df	F	p-value	
s(idade)	1.000	1.000	16.322	9.59e-05	***
s(altura)	5.311	6.426	3.857	0.00115	**
s(peso)	1.000	1.000	22.257	6.56e-06	***
s(fcprep)	1.856	2.337	14.865	8.39e-07	***
---					

R-sq.(adj) = 0.64 Deviance explained = 67.8%

GCV = 3.8663 Scale est. = 3.435 n = 127

## MAG: exemplo

- Como o número efetivo de graus de liberdade para idade e peso é igual a 1, elas se comportam de forma linear no modelo. Os gráficos dispostos na Figura 6, produzidos por meio do comando `plot(mod2, se=TRUE)` evidenciam esse fato; além disso mostram a natureza “mais não linear” da variável altura (com `edf = 5.311`).
- Uma avaliação da qualidade do ajuste pode ser realizada por meio de uma análise de resíduos e de comparação dos valores observados e preditos. Para essa finalidade, o comando `gam.check(mod2)` gera os gráficos apresentados na Figura 7 que não evidenciam problemas no ajuste.
- Além disso, é possível comparar os modelos por meio de uma **análise de desviância**, que pode ser obtida com o comando `anova(mod0, mod1, mod2, test= "F")`.

## MAG: exemplo

## Analysis of Deviance Table

Model 1: vo2fcrico ~ NYHA + idade + altura + peso + fcrep +  
vo2rep

Model 2: vo2fcrico ~ NYHA + s(idade) + s(altura) + s(peso) +  
s(fcrep) + s(vo2rep)

Model 3: vo2fcrico ~ NYHA + s(idade) + s(altura) + s(peso) +  
s(fcrep)

	Resid.	Df	Resid.	Dev	Df	Deviance	F	Pr(>F)
1	117.00		439.24					
2	109.72		383.18	7.2766		56.052	2.2425	0.03404 *
3	111.24		387.58	-1.5129		-4.399	0.8465	0.40336

## MAG: exemplo

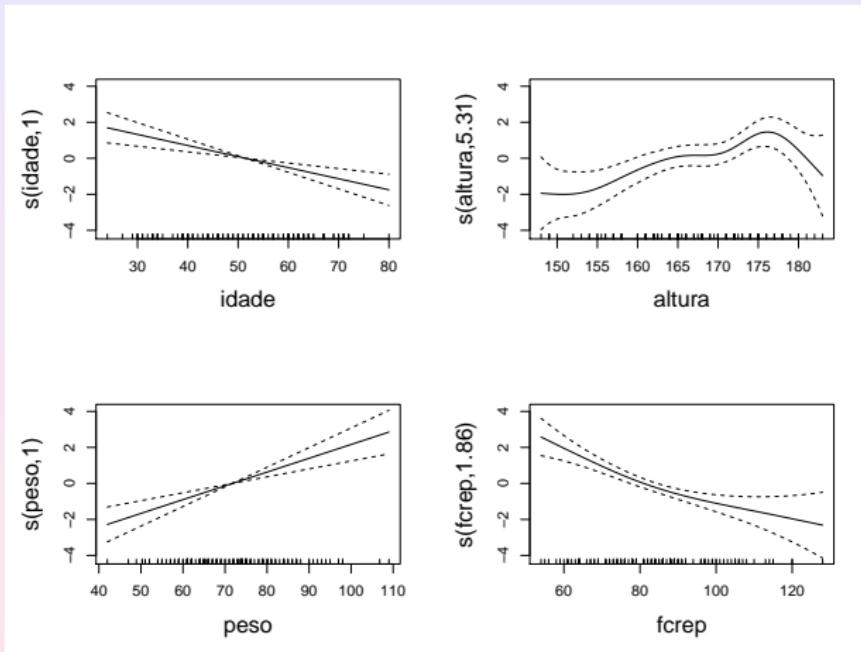


Figura 6: Funções suavizadas (com bandas de confiança) obtidas por meio do modelo GAM para os dados do Exemplo.

## MAG: exemplo

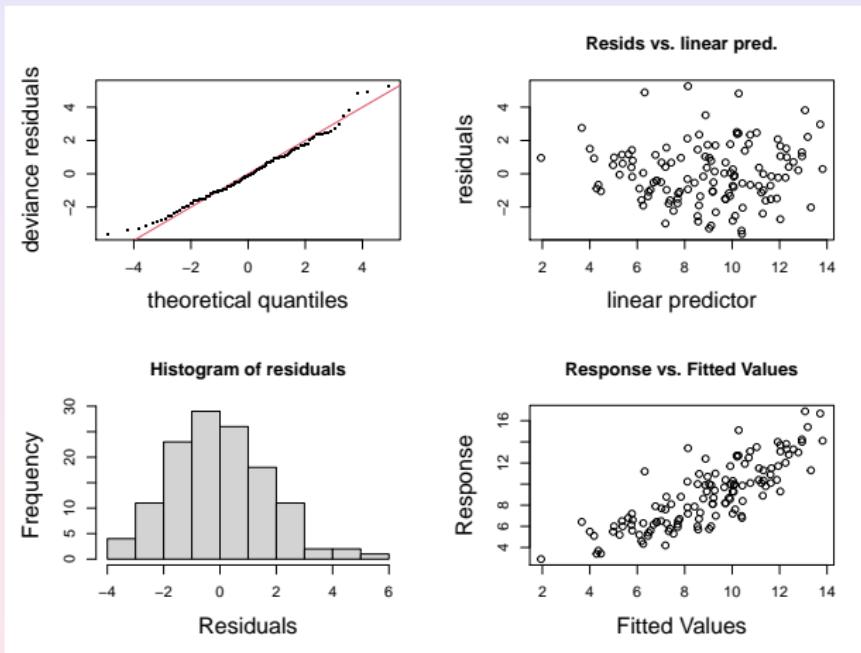


Figura 7: Gráficos diagnósticos para o ajuste do modelo GAM aos dados do Exemplo.

## MAG: exemplo

- Esses resultados mostram que ambos os modelos GAM são essencialmente equivalentes ( $p = 0.403$ ) mas significativamente mais adequados ( $p = 0.034$ ) que o modelo de regressão linear múltipla.
- A previsão para um novo conjunto dados em que apenas os valores das variáveis preditoras estão disponíveis pode ser obtida por meio do comando `predict(mod2, newdata=esforcoprev, se=TRUE, type="response")`. Consideremos, por exemplo, o seguinte conjunto com dados de 5 novos pacientes

idade	altura	peso	NYHA	fcrep	vo2rep
66	159	50	2	86	3,4
70	171	77	4	108	4,8
64	167	56	2	91	2,5
42	150	67	2	70	3,0
54	175	89	2	91	2,9

## MAG: exemplo

O resultado da previsão com o modelo adotado é

```
$fit
  1          2          3          4          5
4.632615  5.945157  5.928703  7.577097 10.273719
$se.fit
  1          2          3          4          5
0.6747203 0.7155702 0.6255449 0.7731991 0.5660150
```

## Referências

- Bühlmann, P. and van de Geer, S. (2011). *Statistics for High-Dimensional Data*. Berlin: Springer.
- Friedman, J. H., Hastie, T. and Tibshirani, R. (2010). Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, **33**, 1–22.
- Hastie, T., Tibshirani, R. and Wainwright, M. (2015). *Statistical Learning with Sparsity*. Chapman and Hall.
- James, G., Witten, D., Hastie, T. and Tibshirani, R. (2017). *Introduction to Statistical Learning*. Springer.
- Morettin, P. A. e Singer, J. M. (2021). *Estatística e Ciência de Dados*. Texto Preliminar, IME-USP.
- Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B (methodological)*, **58**, 267–288.

# MAE 5905: Introdução à Ciência de Dados

Pedro A. Morettin

Instituto de Matemática e Estatística  
Universidade de São Paulo  
[pam@ime.usp.br](mailto:pam@ime.usp.br)  
<http://www.ime.usp.br/~pam>

## Aula 7

14 de abril de 2023

# Sumário

1 Exemplo-Regularização

2 VC e Bootstrap

3 Classificação Clássica

## Regularização - EMQ

- **Exemplo.** Vamos considerar o conjunto de dados **esforço**, centrando o interesse na predição da variável resposta  $Y$ : VO2 (consumo de oxigênio em  $\text{ml}/(\text{kg} \cdot \text{min})$ ) com base nas variáveis preditoras  $X_1$ : Idade (em anos),  $X_2$ : Peso (em kg),  $X_3$ : Superfície corpórea e  $X_4$ : IMC (índice de massa corpórea em  $\text{kg}/\text{m}^2$ ) ( $n = 126$ )
- Ajustando o modelo via mínimos quadrados ordinários, obtemos os coeficientes:

Coefficients	Estimate	Std. Error	t value	P-value
Intercept	14.92204	6.69380	2.229	0.0276 *
Idade	-0.01005	0.02078	-0.483	0.6297
Peso	0.05233	0.11760	0.445	0.6571
Sup.Corp.	-1.40678	6.25353	-0.225	0.8224
IMC	-0.20030	0.16104	-1.244	0.2160

Residual standard error: 2.822 on 121 degrees of freedom

Multiple R-squared: 0.03471, Adjusted R-squared: 0.002799

## Regularização - Ridge

- Os coeficientes correspondentes obtidos por meio de regularização **Ridge** são

Intercept	5.185964640
Idade	-0.000133776
Peso	-0.006946405
Sup.Corp	-0.295094364
IMC	-0.022923850

- O valor do coeficiente de regularização  $\lambda = 0,82065$ , mostra que as estimativas para os coeficientes de Idade e Peso foram encolhidas para zero, enquanto aquelas correspondentes à Sup.Corp tem peso maior do que as demais.
- Neste caso, a raiz quadrada do **erro quadrático médio** (*root mean squared error*) e o **coeficiente de determinação** são, respectivamente  $RMSE = 0,928$  e  $R^2 = 0,235$ .

## Regularização Lasso

- Os coeficientes correspondentes obtidos por meio de regularização Lasso são

Intercept 4.95828012

Idade .

Peso -0.01230145

Sup.Corp .

IMC -0.02011871

- O valor do coeficiente de regularização  $\lambda = 0,0257$  mostra que as estimativas dos coeficientes Idade e Sup. Corp foram efetivamente encolhidas para zero.
- Neste caso RMSE e  $R^2$  são, respectivamente, 0,927 e 0,228.

## Regularização Elastic Net

- Os coeficientes correspondentes obtidos por meio de regularização Elastic Net são:

Intercept	4.985532570
Idade	.
Peso	-0.009099925
Sup.Corp	-0.097034844
IMC	-0.023302254

- Os parâmetros de suavização estimados foram  $\alpha = 0,1$  e  $\lambda = 0,227$  e também indicam que o coeficiente associado à Idade foi encolhido para zero.
- Também obtemos  $RMSE=0,927$  e  $R^2 = 0,228$  neste caso.
- Os três métodos de regularização têm desempenhos similares quando vistos pelas óticas do RMSE e do  $R^2$ .

## Validação Cruzada

- **Validação cruzada** é a denominação atribuída a um conjunto de técnicas utilizadas para avaliar o erro de previsão de modelos estatísticos. O erro de previsão é uma medida da precisão com que um modelo pode ser usado para prever o valor de uma nova observação, ou seja diferente daquelas utilizados para o ajuste do modelo.
- Em modelos de regressão o **erro de previsão** é definido como

$$EP = E(y - \hat{y})^2,$$

em que  $y$  representa uma nova observação e  $\hat{y}$  é a previsão obtida pelo modelo.

- **erro quadrático médio dos resíduos** pode ser usado como uma estimativa do erro de previsão (EP), mas tende, em geral, a ser muito otimista, ou seja, a subestimar o seu verdadeiro valor. Uma razão é que os mesmos dados são utilizados para ajustar e avaliar o modelo.
- No processo de validação cruzada, o modelo é ajustado a um subconjunto dos dados (**subconjunto de treinamento**) e o resultado é empregado num outro subconjunto (**subconjunto de teste**) para avaliar se ele tem um bom desempenho ou não.

# Validação Cruzada

Um algoritmo utilizado nesse processo é o seguinte (Efron e Tibshirani, 1993):

- 1) Dadas  $n$  observações,  $y_1, \dots, y_n$ , o modelo é ajustado  $n$  vezes, em cada uma delas eliminando uma observação e o valor previsto, denotado por  $\hat{y}_{-i}$ , para essa observação, é calculado com base no resultados obtido com as demais  $n - 1$ .
- 2) O erro de previsão é estimado por

$$VC_{(n)} = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_{-i})^2. \quad (1)$$

Esse tipo de validação cruzada é chamado **validação cruzada com eliminação de uma observação** (leave-one-out cross-validation - LOOCV).

# Validação Cruzada

- Na chamada **validação cruzada de ordem  $k$**  ( *$k$ -fold cross validation*) o conjunto de dados original é subdividido em dois, sendo um deles utilizado como conjunto de treinamento e o segundo como conjunto de teste. Esse processo é repetido  $k$  vezes com conjuntos de treinamento e testes diferentes como esquematizado na Figura 5.
- O correspondente erro de previsão é estimado como

$$VC_{(k)} = \frac{1}{k} \sum_{i=1}^k EQM_i. \quad (2)$$

em que

$$EQM_i = \sum (y_j - \hat{y}_j)^2 / n_i$$

é erro quadrático médio obtido no  $i$ -ésimo ajuste,  $i = 1, \dots, k$ . Aqui,  $y_j$ ,  $\hat{y}_j$  e  $n_i$  são, respectivamente, os valores observado e previsto para a  $j$ -ésima observação e o número de observações no  $i$ -ésimo conjunto de teste. O usual é tomar  $k = 5$  ou  $k = 10$ .

## Validação Cruzada

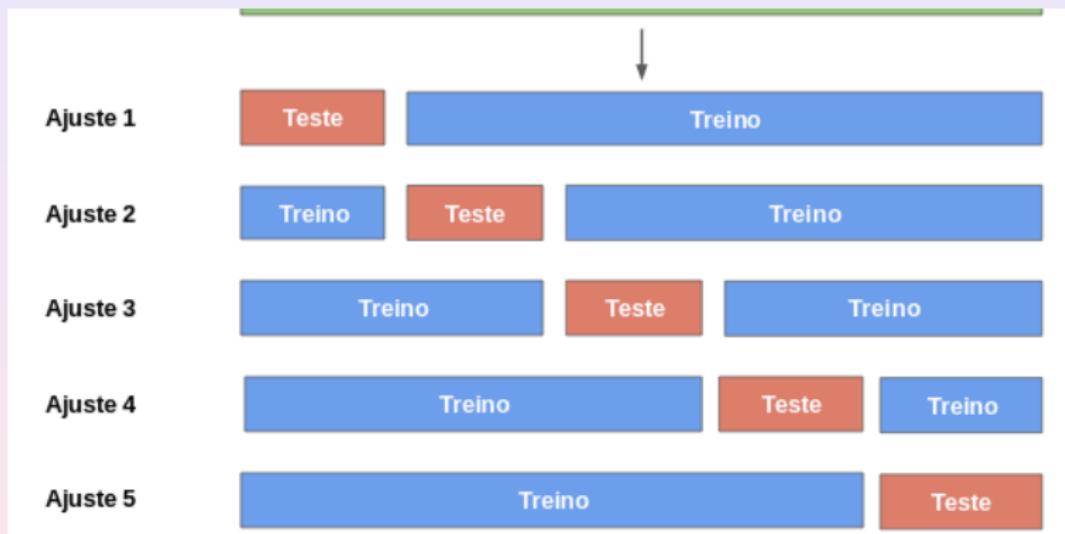


Figura 5: Representação esquemática da divisão dos dados para validação cruzada de ordem  $k$ .

## Bootstrap

- Com o progresso de métodos computacionais e com capacidade cada vez maior de lidar com conjuntos grandes de dados, o cálculo de erros padrões, vieses etc, pode ser feito sem recorrer a uma teoria, que muitas vezes pode ser muito complicada ou simplesmente não existir.
- Um desses métodos é chamado **bootstrap**, introduzido por B. Efrom, em 1979. A ideia básica do método bootstrap é re-amostrar o conjunto disponível de dados para estimar um parâmetro  $\theta$ , com o fim de criar **dados replicados**. A partir dessas replicações, podemos avaliar a variabilidade de um estimador proposto para  $\theta$ , sem recorrer a cálculos analíticos.
- Suponha que temos dados  $\mathbf{x} = (x_1, x_2, \dots, x_n)$  e queremos estimar a mediana populacional,  $M_d$ , por meio da mediana amostral  $md(\mathbf{x}) = med(x_1, \dots, x_n)$ .
- Escolhemos uma amostra aleatória simples, *com reposição*, de tamanho  $n$  dos dados. Tal amostra é chamada uma **amostra bootstrap** e denotada por  $\mathbf{x}^* = (x_1^*, \dots, x_n^*)$ .

## Bootstrap

- Suponha, agora, que geramos  $B$  tais amostras independentes, denotadas  $\mathbf{x}_1^*, \dots, \mathbf{x}_B^*$ . Para cada amostra bootstrap, geramos uma **réplica bootstrap** do estimador proposto, ou seja, de  $md(x)$ , obtendo-se

$$md(\mathbf{x}_1^*), md(\mathbf{x}_2^*), \dots, md(\mathbf{x}_B^*). \quad (3)$$

- Definimos o **estimador bootstrap do erro padrão** de  $md(x)$  como

$$\widehat{e.p.}_B(md) = \left[ \frac{\sum_{b=1}^B (md(\mathbf{x}_b^*) - md(\cdot))^2}{B - 1} \right]^{1/2}, \quad (4)$$

com

$$md(\cdot) = \frac{\sum_{b=1}^B md(\mathbf{x}_b^*)}{B}. \quad (5)$$

Ou seja, o estimador bootstrap do erro padrão da mediana amostral é o desvio padrão amostral do conjunto (3).

## Bootstrap

- A questão que se apresenta é: qual deve ser o valor de  $B$ , ou seja, quantas amostras bootstrap devemos gerar para estimar erros padrões de estimadores? A experiência indica que um valor razoável é  $B = 200$ .
- No caso geral de um estimador  $\hat{\theta} = t(\mathbf{x})$ , o **algoritmo bootstrap** para estimar o erro padrão de  $\hat{\theta}$  é o seguinte:

[1] Selecione  $B$  amostras bootstrap independentes  $\mathbf{x}_1^*, \dots, \mathbf{x}_B^*$ , cada uma consistindo de  $n$  valores selecionados com reposição de  $\mathbf{x}$ . Tome  $B \approx 200$ .

[2] Para cada amostra bootstrap  $\mathbf{x}_b^*$  calcule a réplica bootstrap

$$\hat{\theta}^*(b) = t(\mathbf{x}_b^*), \quad b = 1, 2, \dots, B.$$

[3] O erro padrão de  $\hat{\theta}$  é estimado pelo desvio padrão das  $B$  réplicas:

$$\widehat{\text{e.p.}}_B = \left[ \frac{1}{B-1} \sum_{b=1}^B (\hat{\theta}^*(b) - \hat{\theta}^*(\cdot))^2 \right]^{1/2}, \quad (6)$$

com

$$\hat{\theta}^*(\cdot) = \frac{1}{B} \sum_{b=1}^B \hat{\theta}^*(b). \quad (7)$$

# Classificação

- De modo genérico, vamos designar a variável preditora por  $X$  (que pode ser escalar ou vetorial) e a resposta (indicadora de uma classe) por  $Y$ .
- Os dados serão indicados por  $(x_i, y_i)$ ,  $i = 1, \dots, n$ . A ideia é usar os dados para obter agrupamentos cujos elementos sejam de alguma forma parecidos entre si (com base em alguma medida obtida a partir da variável preditora) e depois utilizar essa medida para classificar um ou mais novos elementos (para os quais dispomos apenas dos valores da variável preditora) em uma das classes.
- Se tivermos  $d$  variáveis preditoras e uma resposta dicotómica (*i.e.*, duas classes), um **classificador** é uma função que mapeia um espaço  $d$ -dimensional sobre  $\{-1, 1\}$ .
- Formalmente, seja  $(X, Y)$  um vetor aleatório, de modo que  $X \in \mathbb{R}^d$  e  $Y \in \{-1, 1\}$ . Então, um classificador é uma função  $g : \mathbb{R}^d \rightarrow \{-1, 1\}$  e a **função erro** ou **risco** é a probabilidade de erro,  $L(g) = P\{g(X) \neq Y\}$ .

# Classificação

- A acurácia de um estimador de  $g$ , digamos  $\hat{g}$ , pode ser medida pelo estimador de  $L(g)$ , chamado de **taxa de erros**, que é a proporção de erros gerados pela aplicação de  $\hat{g}$  às observações do conjunto de dados, ou seja,

$$\widehat{L}(\hat{g}) = \frac{1}{n} \sum_{i=1}^n I(y_i \neq \hat{y}_i), \quad (8)$$

com  $\hat{y}_i = \hat{g}(x_i)$  indicando o rótulo (-1 ou 1) da classe prevista por meio de  $\hat{g}$ . Se  $I(y_i \neq \hat{y}_i) = 0$ , a  $i$ -ésima observação estará classificada corretamente.

- Sob o enfoque de aprendizado automático (AA), o objetivo é comparar diferentes modelos para identificar aquele com menor taxa de erros.

# Classificação

- Nesse contexto, dispomos de um conjunto de **dados de treinamento**  $(x_i, y_i)$ ,  $i = 1, \dots, n$  e de um conjunto de dados de **teste**, cujo elemento típico é  $(x_0, y_0)$ . O interesse é minimizar a **taxa de erro de teste** associada ao conjunto de observações teste que pode ser estimada por

$$\text{Média}[I(y_0 \neq \hat{y}_0)], \quad (9)$$

em que a média é calculada relativamente aos elementos do conjunto de dados de teste.

- O classificador (ou modelo) ótimo é aquele que minimiza (9).
- Com o objetivo de classificar os elementos do conjunto de dados, deve-se ajustar o classificador ótimo ao conjunto de dados disponíveis (treinamento e teste) e utilizar a estimativa  $\hat{g}$  daí obtida para classificar os elementos do conjunto de dados para classificação.
- Quando dispomos de apenas um conjunto de dados, podemos recorrer ao processo de validação cruzada para dividi-lo em conjuntos de dados de treinamento e de dados de teste.

# Classificação clássica

Neste contexto, podemos ter:

- 1) Classificação por regressão logística;
- 2) Classificação bayesiana;
- 3) Função discriminante linear de Fisher;
- 4) Classificador K-vizinho mais próximo.
- 5) Outras propostas

## Classificação por regressão logística - RL

- **Exemplo.** Os dados da Tabela 1 são extraídos de um estudo realizado no Hospital Universitário da Universidade de São Paulo com o objetivo de avaliar se algumas medidas obtidas ultrassonograficamente poderiam ser utilizadas como substitutas de medidas obtidas por métodos de ressonância magnética, considerada como padrão áureo, para avaliação do deslocamento do disco da articulação temporomandibular (doravante referido simplesmente como disco).
- Distâncias cápsula-côndilo (em mm) com boca aberta ou fechada (referidas, respectivamente, como distância aberta ou fechada no restante do texto) foram obtidas ultrassonograficamente de 104 articulações e o disco correspondente foi classificado como deslocado (1) ou não (0) segundo a avaliação por ressonância magnética. A variável resposta é o *status* do disco (1 = deslocado ou 0 = não).
- Consideraremos um modelo logístico para a chance de deslocamento do disco, tendo apenas a distância aberta como variável explicativa: Nesse contexto, o modelo

$$\log[\theta(x_i; \alpha, \beta)]/[1 - \theta(x_i; \alpha, \beta)] = \alpha + x_i\beta \quad (10)$$

$$i = 1, \dots, 104.$$

# RL-Exemplo

**Tabela:** Parte dos Dados de um estudo odontológico

Dist aberta	Dist fechada	Desloc disco	Dist aberta	Dist fechada	Desloc disco	Dist aberta	Dist fechada	Desloc disco
2.2	1.4	0	0.9	0.8	0	1.0	0.6	0
2.4	1.2	0	1.1	0.9	0	1.6	1.3	0
2.6	2.0	0	1.4	1.1	0	4.3	2.3	1
3.5	1.8	1	1.6	0.8	0	2.1	1.0	0
1.3	1.0	0	2.1	1.3	0	1.6	0.9	0
2.8	1.1	1	1.8	0.9	0	2.3	1.2	0
1.5	1.2	0	2.4	0.9	0	2.4	1.3	0
2.6	1.1	0	2.0	2.3	0	2.0	1.1	0
1.2	0.6	0	2.0	2.3	0	1.8	1.2	0
1.7	1.5	0	2.4	2.9	0	1.4	1.9	0
1.3	1.2	0	2.7	2.4	1	1.5	1.3	0
1.2	1.0	0	1.9	2.7	1	2.2	1.2	0
4.0	2.5	1	2.4	1.3	1	1.6	2.0	0
1.2	1.0	0	2.1	0.8	1	1.5	1.1	0
3.1	1.7	1	0.8	1.3	0	1.2	0.7	0
2.6	0.6	1	0.8	2.0	1	1.5	0.8	0
1.8	0.8	0	0.5	0.6	0	1.8	1.1	0
1.2	1.0	0	1.5	0.7	0	2.3	1.6	1
1.9	1.0	0	2.9	1.6	1	1.2	0.4	0
1.2	0.9	0	1.4	1.2	0	1.0	1.1	0
1.7	0.9	1	3.2	0.5	1	2.9	2.4	1
1.2	0.8	0	1.2	1.2	0	2.5	3.3	1

Dist aberta: distância cápsula-côndilo com boca aberta (mm)

Dist fechada: distância cápsula-côndilo com boca fechada (mm)

Desloc disco: deslocamento do disco da articulação temporomandibular (1=sim, 0=não)

## RL–Exemplo

- No modelo,  $\theta(x_i; \alpha, \beta)$  representa a probabilidade de deslocamento do disco quando o valor da distância aberta é  $x_i$ ,  $\alpha$  denota o logaritmo da chance de deslocamento do disco quando a distância aberta tem valor  $x_i = 0$  e  $\beta$  é interpretado como a variação no logaritmo da chance de deslocamento do disco por unidade de variação da distância aberta.
- Consequentemente, a razão de chances do deslocamento do disco correspondente a uma diferença de  $d$  unidades da distância aberta será  $\exp(d \times \beta)$ . Como não temos dados correspondentes a distâncias abertas menores que 0,5, convém substituir os valores  $x_i$  por valores “centrados”, ou seja por  $x_i^* = x_i - x_0$ .
- Uma possível escolha para  $x_0$  é o mínimo de  $x_i$ , que é 0,5. Essa transformação na variável explicativa altera somente a interpretação do parâmetro  $\alpha$  que passa a ser o logaritmo da chance de deslocamento do disco quando a distância aberta tem valor  $x_i = 0,5$ .

## RL–Exemplo–Uso do R

Call:

```
glm(formula = deslocamento ~ (distanciaAmin), family = binomial,  
    data = disco)
```

Deviance Residuals:

Min	1Q	Median	3Q	Max
-1.5240	-0.4893	-0.3100	0.1085	3.1360

Coefficients:

	Estimate	Std. Error	z value	Pr(> z )
(Intercept)	-5.8593	1.1003	-5.325	1.01e-07 ***
distanciaAmin	3.1643	0.6556	4.827	1.39e-06 ***
---				

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 123.11 on 103 degrees of freedom  
Residual deviance: 71.60 on 102 degrees of freedom  
AIC: 75.6

Number of Fisher Scoring iterations: 6

## RL-Exemplo

- Estimativas (com erros padrões entre parênteses) dos parâmetros desse modelo ajustado por máxima verossimilhança aos dados da Tabela 1, são,  $\hat{\alpha} = -5,86 (1,10)$  e  $\hat{\beta} = 3,16 (0,66)$  e então, segundo o modelo, uma estimativa da chance de deslocamento do disco para articulações com distância aberta  $x = 0,5$  (que corresponde à distância aberta transformada  $x^* = 0,0$ ) é  $\exp(-5,86) = 0,003$ .
- Um intervalo de confiança (95%) para essa chance pode ser obtido exponenciando os limites ( $LI$  e  $LS$ ) do intervalo para o parâmetro  $\alpha$ , nomeadamente,

$$LI = \exp[\hat{\alpha} - 1,96EP(\hat{\alpha})] = \exp(-5,86 - 1,96 \times 1,10) = 0,000$$

$$LS = \exp[\hat{\alpha} + 1,96EP(\hat{\alpha})] = \exp(-5,86 + 1,96 \times 1,10) = 0,025.$$

- Os limites de um intervalo de confiança para a razão de chances correspondentes a um variação de uma unidade no valor da distância aberta podem ser obtidos de maneira similar e são 6,55 e 85,56.
- Substituindo os parâmetros  $\alpha$  e  $\beta$  por suas estimativas  $\hat{\alpha}$  e  $\hat{\beta}$  em (10) podemos estimar a probabilidade de sucesso (deslocamento do disco, no exemplo sob investigação).

## RL–Exemplo

- Por exemplo, para uma articulação cuja distância aberta seja 2,1 (correspondente à distância aberta transformada igual a 1,6), a estimativa dessa probabilidade é

$$\hat{\theta} = \exp(-5,86 + 3,16 \times 1,6) / [1 + \exp(-5,86 + 3,16 \times 1,6)] = 0,31.$$

- Lembrando que o objetivo do estudo é substituir o processo de identificação de deslocamento do disco realizado via ressonância magnética por aquele baseado na medida da distância aberta por meio de ultrassonografia, podemos estimar as probabilidades de sucesso para todas as articulações e identificar um **ponto de corte**  $d_0$  segundo o qual, distâncias abertas com valores acima dele sugerem decidirmos pelo deslocamento do disco e distâncias abertas com valores abaixo dele sugerem a decisão oposta.
- Obviamente, não esperamos que todas as decisões tomadas dessa forma sejam corretas e consequentemente, a escolha do ponto de corte deve ser feita com o objetivo de minimizar os erros (decidir pelo deslocamento quando ele não existe ou *vice versa*).

## RL–Exemplo

Nesse contexto, um contraste entre as decisões tomadas com base em um determinado ponto de corte  $d_0$  e o padrão áureo definido pela ressonância magnética para todas as 104 articulações pode ser resumido por meio da Tabela 2, em que as frequências da diagonal principal correspondem a decisões corretas e aquelas da diagonal secundária às decisões erradas.

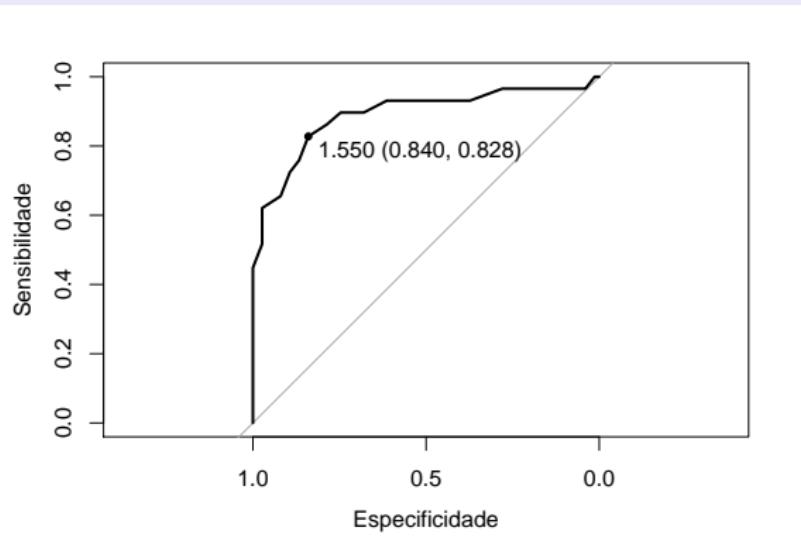
Tabela: Frequência de decisões para um ponto de corte  $d_0$

		Deslocamento real do disco	
		sim	não
Decisão baseada na distância aberta $d_0$	sim	$n_{11}$	$n_{12}$
	não	$n_{21}$	$n_{22}$

## RL–Exemplo

- O quociente  $n_{11}/(n_{11} + n_{21})$  é conhecido como **sensibilidade** do processo de decisão e é uma estimativa da probabilidade de decisões corretas quando o disco está realmente deslocado.
- O quociente  $n_{22}/(n_{12} + n_{22})$  é conhecido como **especificidade** do processo de decisão e é uma estimativa da probabilidade de decisões corretas quando o disco realmente não está deslocado. A situação ideal é aquela em que tanto a sensibilidade quanto a especificidade do processo de decisão são iguais a 100%.
- O problema a resolver é determinar o ponto de corte  $d_{max}$  que gere o melhor equilíbrio entre sensibilidade e especificidade. Com essa finalidade, podemos construir tabelas com o mesmo formato da Tabela 2 para diferentes pontos de corte e um gráfico cartesiano entre a sensibilidade e especificidade obtida de cada uma delas.
- Esse gráfico, conhecido como **curva ROC** (do termo inglês **Receiver Operating Characteristic**) gerado para os dados da Tabela 1 está apresentado na Figura 1.

## RL–Exemplo



**Figura:** Curva ROC para os dados da Tabela 1 baseada no modelo (10) com distância aberta como variável explicativa.

## RL-Exemplo

- O ponto de corte ótimo é aquele mais próximo do vértice superior esquerdo (em que tanto a sensibilidade quanto a especificidade seriam iguais a 100%).
- Para o exemplo, esse ponto está salientado na Figura 1 e corresponde à distância aberta com valor  $d_{max} = 2,05 (= 1,55 + 0,5)$ . A sensibilidade e a especificidade associadas à decisão baseada nesse ponto de corte, são, respectivamente, 83% e 84% e as frequências de decisões corretas estão indicadas na Tabela 3.

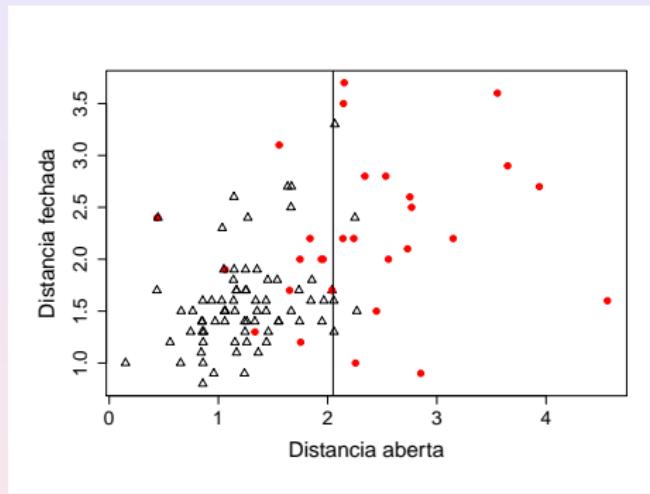
**Tabela:** Frequência de decisões para um ponto de corte para distância aberta  $d_{max} = 2,05$

	Deslocamento real do disco	
	sim	não
Decisão baseada na distância aberta $d_{max} = 2,05$	sim	24
	não	5
		63

## RL–Exemplo

- Com esse procedimento de decisão a porcentagem de acertos (**acurácia**) é 84% [=  $(24 + 63)/104$ ]. A porcentagem de **falsos positivos** é 17% [=  $5/(5 + 29)$ ] e a porcentagem de **falsos negativos** é 16% [=  $12/(12 + 63)$ ].
- Um gráfico de dispersão com o correspondente ponto de corte baseado apenas na distância aberta está apresentado na Figura 2 com símbolos vermelhos indicando casos com deslocamento do disco e em preto indicando casos sem deslocamento. Os valores de ambas as distâncias foram ligeiramente alterados para diminuir a superposição nos pontos.

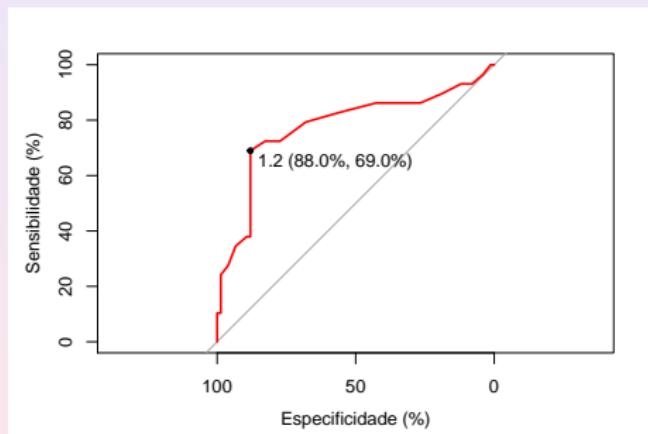
## RL–Exemplo



**Figura:** Gráfico de dispersão para os dados da Tabela 1 com ponto de corte baseado apenas na distância aberta.

## RL–Exemplo

Uma análise similar, baseada na distância fechada (transformada por meio da subtração de seu valor mínimo, 0,4) gera a curva ROC apresentada na Figura 3 e frequências de decisões apresentada na Tabela 4.



**Figura:** Curva ROC para os dados da Tabela 1 baseada no modelo (10) com distância fechada como variável explicativa.

## RL–Exemplo

**Tabela:** Frequência de decisões para um ponto de corte para distância fechada  
 $d_{max} = 1,6$

Decisão baseada na distância fechada $d_{max} = 1,6$	Deslocamento real do disco	
	sim	não
sim	20	9
não	9	66

## RL-Exemplo

- A acurácia associada a processo de decisão baseado apenas na distância fechada, 83% [=  $(20 + 66)/104$ ] é praticamente igual àquela obtida com base apenas na distância aberta; no entanto aquele processo apresenta um melhor equilíbrio entre sensibilidade e especificidade (83% e 84%, respectivamente, *versus* 69% e 88%).
- Se quisermos avaliar o processo de decisão com base nas observações das distâncias aberta e fechada simultaneamente, podemos considerar o modelo

$$\log[\theta(x_i; \alpha, \beta, \gamma)]/[1 - \theta(x_i; \alpha, \beta, \gamma)] = \alpha + x_i\beta + w_i\gamma \quad (11)$$

$i = 1, \dots, 104$  em que  $w_i$  corresponde à distância fechada observada na  $i$ -ésima articulação.

## RL–Exemplo

- Neste caso,  $\gamma$  corresponde à razão entre a chance de deslocamento do disco para articulações com distância fechada  $w + 1$  e a chance de deslocamento do disco para articulações com distância fechada  $w$  para aquelas com mesmo valor da distância aberta; uma interpretação similar vale para o parâmetro  $\beta$ .
- Estimativas dos parâmetros (com erros padrões entre parênteses) do modelo (11) obtidas após a transformação das variáveis explicativas segundo o mesmo figurino adotado nas análises univariadas são  $\hat{\alpha} = -6,38 (1,19)$ ,  $\hat{\beta} = 2,83 (0,67)$  e  $\hat{\gamma} = 0,98 (0,54)$ .
- A estimativa do parâmetro  $\gamma$  é apenas marginalmente significativa, ou seja a inclusão da variável explicativa distância fechada não acrescenta muito poder de discriminação além daquele correspondente à distância aberta.

## RL–Exemplo

- Uma das razões para isso é que as duas variáveis são correlacionadas (com coeficiente de correlação de Pearson igual a 0,46). A determinação de pontos de corte para modelos com duas ou mais variáveis explicativas é bem mais complexa do que no caso univariado e não será abordada neste texto.
- Para efeito de comparação com as análises anteriores, as frequências de decisões obtidas com os pontos de corte utilizados naquelas estão dispostas na Tabela 5, e correspondem a uma sensibilidade de 62%, especificidade de 97% e acurácia de 88%.

**Tabela:** Frequência de decisões correspondentes a pontos de corte  $d_{max} = 2,05$  para distância aberta e  $d_{max} = 1,6$  para distância fechada

Decisão baseada em ambas as distâncias	Deslocamento real do disco	
	sim	não
sim	18	2
não	11	73

## RL–Exemplo

- Numa segunda análise, agora sob o paradigma de aprendizado automático (AA), a escolha do modelo ótimo é baseada apenas nas porcentagens de classificação correta (acurácia) obtidas por cada modelo num conjunto de dados de teste a partir de seu ajuste a um conjunto de dados de treinamento. Como neste caso não dispomos desses conjuntos *a priori*, podemos recorrer à técnica de **validação cruzada**.
- Neste exemplo, utilizamos validação cruzada de ordem 5 com 5 repetições (VC5/5), em que o conjunto de dados é dividido em dois, cinco vezes, gerando cinco conjuntos de dados de treinamento e de teste. A análise é repetida cinco vezes em cada conjunto e a acurácia média obtida das 25 análises serve de base para a escolha do melhor modelo.
- Comparamos quatro modelos de regressão logística, os dois primeiros com apenas uma das variáveis preditoras (distância aberta ou distância fechada), o terceiro com ambas incluídas aditivamente e o último com ambas as distâncias e sua interação. A análise pode ser concretizada por meio do pacote **caret**.

## RL–Exemplo

Os resultados estão dispostos na Tabela 6 tanto para validação cruzada VC5/5 quanto para validação cruzada LOOCV.

**Tabela:** Acurácia obtida por validação cruzada para as regressões logísticas ajustados as dados do Exemplo 8.1

Modelo	Variáveis	Acurácia VC5/5	Acurácia LOOCV
1	Distância aberta	84,8 %	84,6 %
2	Distância fechada	75,2 %	74,0 %
3	Ambas (aditivamente)	85,7 %	85,6 %
4	Ambas + Interação	83,6 %	83,6 %

## RL–Exemplo

- Com ambos os critérios, o melhor modelo é aquele que inclui as duas variáveis preditoras de forma aditiva. Para efeito de classificar uma nova observação (para a qual só dispomos dos valores das variáveis preditoras, o modelo selecionado deve ser ajustado ao conjunto de dados original (treinamento + teste) para obtenção dos coeficientes do classificador.
- Os comandos e a saída associada ao ajuste desse modelo aos 5 conjuntos de dados gerados para validação cruzada e no conjunto completo seguem.
- A seleção obtida por meio do AA corresponde ao modelo (11). Embora a variável Distância fechada seja apenas marginalmente significativa, sua inclusão aumenta a proporção de acertos (acurácia) de 84% no modelo que inclui apenas Distância aberta para 86%.
- A estatística Kappa apresentada juntamente com a acurácia serve para avaliar a concordância entre o processo de classificação e a classificação observada (veja a Seção 4.2).

## RL-Exemplo

```
> set.seed(369321)
> train_control =
      trainControl(method="repeatedcv", number=5, repeats=5)
> model3 = train(deslocamento ~ distanciaAmin + distanciaFmin,
                  data=disco, method="glm", family=binomial,
                  trControl=train_control)
> model3
Generalized Linear Model

104 samples
  2 predictor
  2 classes: 0, 1

No pre-processing
Resampling: Cross-Validated (5 fold, repeated 5 times)
Summary of sample sizes: 83, 83, 84, 83, 83, ...
Resampling results:

  Accuracy   Kappa
0.8573333 0.6124102

> disco$predito3 = predict(model3, newdata=disco, type="raw")
> summary(model3$finalModel)
```

## RL-Exemplo

Deviance Residuals:

Min	1Q	Median	3Q	Max
-1.82771	-0.45995	-0.28189	0.07403	2.82043

Coefficients:

	Estimate	Std. Error	z value	Pr(> z )
(Intercept)	-6.3844	1.1932	-5.351	8.76e-08 ***
distanciaAmin	2.8337	0.6676	4.245	2.19e-05 ***
distanciaFmin	0.9849	0.5383	1.830	0.0673 .

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 123.107 on 103 degrees of freedom  
Residual deviance: 67.991 on 101 degrees of freedom  
AIC: 73.991  
Number of Fisher Scoring iterations: 6

```
> table(disco$deslocamento, disco$predito3)
  0   1
0 72  3
1 12 17
```

Como a divisão do conjunto original nos subconjuntos de treinamento e de teste envolve uma escolha aleatória, os resultados podem diferir (em geral de forma desprezável) para diferentes aplicações dos mesmos comandos, a não ser que se especifique a semente do processo aleatório de divisão por meio do comando `set.seed()`.

## Referências

- Friedman, J. H., Hastie, T. and Tibshirani, R. (2010). Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, **33**, 1–22.
- Hastie, T., Tibshirani, R. and Wainwright, M. (2015). *Statistical Learning with Sparsity*. Chapman and Hall.
- James, G., Witten, D., Hastie, T. and Tibshirani, R. (2017). *Introduction to Statistical Learning*. Springer.
- Morettin, P. A. e Singer, J. M. (2021). *Estatística e Ciência de Dados*. Texto Preliminar, IME-USP.

# MAE 5905: Introdução à Ciência de Dados

Pedro A. Morettin

Instituto de Matemática e Estatística  
Universidade de São Paulo  
[pam@ime.usp.br](mailto:pam@ime.usp.br)  
<http://www.ime.usp.br/~pam>

## Aula 8

24 de abril de 2023

# Sumário

## 1 Análise discriminante linear

- Classificador de Bayes
- Classificador de Fisher
- Classificador Vizinho mais Próximo - KNN

## 2 Outras Propostas

# ADL

Podemos ter:

- 1) Classificador de Bayes
- 2) Classificador linear de Fisher
- 3) Classificador do vizinho mais próximo
- 4) Outras propostas

## ADL: classificador de Bayes

- Consideremos um conjunto de dados,  $(\mathbf{x}_i, y_i)$ ,  $i = 1, \dots, n$ , em que  $\mathbf{x}_i$  representa os valores de  $p$  variáveis preditoras (explicativas) e  $y_i$  representa o valor de uma variável resposta indicadora da classe a que o  $i$ -ésimo elemento desse conjunto pertence.
- Seja  $\pi_k$  a probabilidade *a priori* de que um elemento com valor das variáveis preditoras  $\mathbf{x} = (x_1, \dots, x_p)$  pertença à classe  $C_k$ ,  $k = 1, \dots, K$  e seja  $f_k(\mathbf{x})$  a função densidade de probabilidade da variável preditora  $\mathbf{X}$  para valores  $\mathbf{x}$  associados a elementos dessa classe. Por um abuso de notação escrevemos  $f_k(\mathbf{x}) = P(\mathbf{X} = \mathbf{x} | Y = k)$  (que a rigor só vale no caso discreto).
- Pelo teorema de Bayes,

$$P(Y = k | \mathbf{X} = \mathbf{x}) = p_k(\mathbf{x}) = \frac{\pi_k f_k(\mathbf{x})}{\sum_{\ell=1}^K \pi_\ell f_\ell(\mathbf{x})}, \quad k = 1, \dots, K, \quad (1)$$

é a probabilidade a posteriori de que um elemento com valor das variáveis preditoras igual a  $\mathbf{x}$  pertença à  $k$ -ésima classe. Para calcular essa probabilidade é necessário conhecer  $\pi_k$  e  $f_k(\mathbf{x})$ ; em muitos casos, supõe-se que para os elementos da  $k$ -ésima classe, os valores de  $\mathbf{X}$  tenham uma distribuição  $N_p(\boldsymbol{\mu}_k, \boldsymbol{\Sigma})$ , ou seja, com média que depende da classe  $k$  e matriz de covariâncias comum a todas as classes.

## ADL: classificador de Bayes

- Suponha que  $\mathbf{X} \in \mathcal{X} \subset \mathbb{R}^p$ . Uma **regra de classificação**,  $R$ , consiste em dividir  $\mathcal{X}$  em  $K$  regiões disjuntas  $\mathcal{X}_1, \dots, \mathcal{X}_K$ , tal que se  $\mathbf{x} \in \mathcal{X}_k$ , o elemento correspondente é classificado em  $C_k$ .
- A probabilidade (condicional) de classificação incorreta, i.e., de classificar um elemento com valor das variáveis preditoras  $\mathbf{x}$  em  $C_k$ , quando de fato ele pertence a  $C_j$ ,  $j \neq k$  usando a regra  $R$ , é

$$p(C_k | C_j, R) = \int_{\mathcal{X}_k} f_j(\mathbf{x}) d\mathbf{x}. \quad (2)$$

Se  $k = j$  em (2), obtemos a probabilidade de classificação correta do elemento com valor das variáveis preditoras  $\mathbf{x}$  em  $C_k$ .

## ADL: classificador de Bayes

- Em muitos casos é possível incluir um **custo** de classificação incorreta, denotado por  $Q(C_k|C_j)$  no procedimento de classificação. Usualmente, esses custos não são iguais e admite-se que  $Q(C_k|C_k) = 0$ ,  $k = 1, \dots, K$ .
- O custo médio de classificação incorreta segundo a regra  $R$  é dado por

$$\delta(\mathbf{x}) = \sum_{k=1}^K \pi_k \left[ \sum_{j=1, j \neq k}^K p(C_j|C_k, R) Q(C_j|C_k) \right]. \quad (3)$$

- O **Classificador de Bayes** é obtido por meio da minimização desse custo médio, supondo os custos de classificação incorreta iguais, ou seja, o elemento com valor das variáveis preditoras  $\mathbf{x}$  deve ser classificado em  $C_k$ , se

$$\delta_k(\mathbf{x}) = \sum_{j=1, j \neq k}^K \pi_j f_j(\mathbf{x}) \quad (4)$$

for mínima,  $k = 1, \dots, K$ .

## ADL: classificador de Bayes

- Minimizar (4) é equivalente a classificar  $\mathbf{x}$  em  $C_k$  se

$$\pi_k f_k(\mathbf{x}) = \max_{1 \leq j \leq K} [\pi_j f_j(\mathbf{x})], \quad (5)$$

pois devemos excluir a  $k$ -ésima parcela de (4) que seja máxima, relativamente a todas as possíveis exclusões de parcelas.

- Em particular, se  $K = 2$ , elementos com valor das variáveis preditoras igual a  $\mathbf{x}$  devem ser classificados em  $C_1$  se

$$\frac{f_1(\mathbf{x})}{f_2(\mathbf{x})} \geq \frac{\pi_2}{\pi_1}, \quad (6)$$

e em  $C_2$ , caso contrário. Veja Johnson e Wichern (1998) e Ferreira (2011), para detalhes.

## ADL: classificador de Bayes

- Suponha o caso  $K = 2$  com variáveis  $\mathbf{x}$  seguindo distribuições normais, com médias  $\mu_1$  para elementos da classe  $C_1$ ,  $\mu_2$  para elementos da classe  $C_2$  e matriz de covariâncias  $\Sigma$  comum.

Usando (1), obtemos

$$P(Y = k | \mathbf{X} = \mathbf{x}) = \frac{\pi_k \exp\{-(\mathbf{x} - \mu_k)^\top \Sigma^{-1}(\mathbf{x} - \mu_k)/2\}}{\sum_{\ell=1}^K \pi_\ell \exp\{-(\mathbf{x} - \mu_\ell)^\top \Sigma^{-1}(\mathbf{x} - \mu_\ell)/2\}}, \quad k = 1, 2. \quad (7)$$

- Então, elementos com valores das variáveis preditoras iguais a  $\mathbf{x}$  são classificados em  $C_1$  se

$$\mathbf{d}^\top \mathbf{x} = (\mu_1 - \mu_2)^\top \Sigma^{-1} \mathbf{x} \geq \frac{1}{2} (\mu_1 - \mu_2)^\top \Sigma^{-1} (\mu_1 + \mu_2) + \log(\pi_2/\pi_1). \quad (8)$$

em que  $\mathbf{d} = \Sigma^{-1}(\mu_1 - \mu_2)$  contém os coeficientes da função discriminante.

## ADL: classificador de Bayes

- No caso geral ( $K \geq 2$ ), o classificador de Bayes associa um elemento com valor das variáveis preditoras igual a  $\mathbf{x}$  à classe para a qual

$$\delta_k(\mathbf{x}) = \boldsymbol{\mu}_k^\top \boldsymbol{\Sigma}^{-1} \mathbf{x} - \frac{1}{2} \boldsymbol{\mu}_k^\top \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_k + \log \pi_k \quad (9)$$

for máxima.

- Em particular, para  $p = 1$ , devemos maximizar

$$\delta_k(\mathbf{x}) = \mathbf{x} \frac{\boldsymbol{\mu}_k}{\sigma^2} - \frac{\boldsymbol{\mu}_k^2}{2\sigma^2} + \log \pi_k. \quad (10)$$

- Quando há apenas duas classes,  $C_1$  e  $C_2$ , um elemento com valor da variável preditora igual a  $x$  deve ser classificado na classe  $C_1$  se

$$dx = \frac{\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2}{\sigma^2} x \geq \frac{\boldsymbol{\mu}_1^2 - \boldsymbol{\mu}_2^2}{2\sigma^2} + \log \frac{\pi_2}{\pi_1} \quad (11)$$

e na classe  $C_2$  em caso contrário.

## ADL: classificador de Bayes

- As fronteiras de Bayes [valores de  $\mathbf{x}$  para os quais  $\delta_k(\mathbf{x}) = \delta_\ell(\mathbf{x})$ ] são obtidas como soluções de

$$\boldsymbol{\mu}_k^\top \boldsymbol{\Sigma}^{-1} \mathbf{x} - \frac{1}{2} \boldsymbol{\mu}_k^\top \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_k + \log \pi_k = \boldsymbol{\mu}_\ell^\top \boldsymbol{\Sigma}^{-1} \mathbf{x} - \frac{1}{2} \boldsymbol{\mu}_\ell^\top \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_\ell + \log \pi_\ell, \quad (12)$$

para  $k \neq \ell$ .

- No paradigma bayesiano, os termos utilizados para cálculo das probabilidades *a posteriori* (1) são conhecidos, o que na prática não é realista. No caso  $p = 1$  pode-se aproximar o classificador de Bayes substituindo  $\pi_k$ ,  $\boldsymbol{\mu}_k$ ,  $k = 1, \dots, K$  e  $\sigma^2$  pelas estimativas

$$\hat{\pi}_k = n_k / n$$

em que  $n_k$  corresponde ao número dos  $n$  elementos do conjunto de dados de treinamento pertencentes à classe  $k$ ,

$$\bar{x}_k = \frac{1}{n_k} \sum_{i:y_i=k} x_i, \quad \text{e} \quad S^2 = \frac{1}{n-K} \sum_{k=1}^K \sum_{i:y_i=k} (x_i - \hat{\mu}_k)^2.$$

## ADL: classificador de Bayes

- Com esses estimadores, a fronteira de decisão de Bayes corresponde a solução de

$$(\bar{x}_1 - \bar{x}_2)x = (\bar{x}_1^2 - \bar{x}_2^2)/2 + [\log(\hat{\pi}_2/\hat{\pi}_1)]S^2. \quad (13)$$

- No caso  $K = 2$  e  $p \geq 2$ , os parâmetros  $\mu_1$ ,  $\mu_2$  e  $\Sigma$  são desconhecidas e têm que ser estimadas a partir de amostras das variáveis preditoras associadas aos elementos de  $C_1$  e  $C_2$ . Com os dados dessas amostras, podemos obter estimativas  $\bar{x}_1$ ,  $\bar{x}_2$ ,  $S_1$  e  $S_2$ , das respectivas médias e matrizes de covariâncias.
- Uma estimativa não enviesada da matriz de covariâncias comum  $\Sigma$  é

$$\mathbf{S} = \frac{(n_1 - 1)\mathbf{S}_1 + (n_2 - 1)\mathbf{S}_2}{n_1 + n_2 - 2}. \quad (14)$$

- Quando  $\pi_1 = \pi_2$ , elementos com valor das variáveis preditoras igual a  $\mathbf{x}$  são classificados em  $C_1$  se

$$\hat{\mathbf{d}}^\top \mathbf{x} = (\bar{x}_1 - \bar{x}_2)^\top \mathbf{S}^{-1} \mathbf{x} \geq \frac{1}{2} (\bar{x}_1 - \bar{x}_2)^\top \mathbf{S}^{-1} (\bar{x}_1 + \bar{x}_2) \quad (15)$$

com  $\hat{\mathbf{d}} = \mathbf{S}^{-1}(\bar{x}_1 - \bar{x}_2)$ .

## Classificador de Bayes: exemplo 1

**Exemplo 1.** Suponha que  $f_1(x)$  seja a densidade de uma distribuição normal padrão e  $f_2(x)$  seja a densidade de uma distribuição normal com média 2 e variância 1. Supondo  $\pi_1 = \pi_2 = 1/2$ , elementos com valor das variáveis preditoras igual a  $x$  são classificados em  $\mathcal{C}_1$  se  $f_1(x)/f_2(x) \geq 1$  o que equivale a

$$\frac{f_1(x)}{f_2(x)} = e^{-x^2/2} e^{(x-2)^2/2} \geq 1,$$

ou seja, se  $x \leq 1$ . Consequentemente, as duas probabilidades de classificação incorretas são iguais a 0,159.

## Classificador de Bayes: exemplo 2

**Exemplo 2:** Consideremos os dados do arquivo [inibina](#), analisados por meio de regressão logística no Exemplo 6.9. Um dos objetivos é classificar as pacientes como tendo resposta positiva ou negativa ao tratamento com inibina com base na variável preditora `difinib = inibpos-inibpre`. Das 32 pacientes do conjunto de dados, 59,4% apresentaram resposta positiva (classe  $C_1$ ) e 40,5% apresentaram resposta negativa (classe  $C_2$ ).

Estimativas das médias das duas classes são, respectivamente,  $\bar{x}_1 = 202,7$  e  $\bar{x}_0 = 49,0$ .

Estimativas das correspondentes variâncias são  $S_1^2 = 31630,5$  e  $S_0^2 = 2852,8$  e uma estimativa da variância comum é  $S^2 = (18 \times S_1^2 + 12 \times S_0^2)/30 = 20119,4$ .

De (11) obtemos o coeficiente da função discriminante  $d = (\bar{x}_1 - \bar{x}_0)/S^2 = 0,0076$ . Para decidir em que classe uma paciente com valor de `difinib = x` deve ser alocada, devemos comparar  $d$  com  $(\bar{x}_1^2 - \bar{x}_0^2)/(2S^2) + [\log(\hat{\pi}_0/\hat{\pi}_1)] = 0,58191$ .

Esses resultados podem ser concretizada por meio da função [lda\(\)](#) do pacote **MASS**.

## Classificador de Bayes: exemplo 2

```
lda(inibina$resposta ~ inibina$difinib, data = inibina)
Prior probabilities of groups:
negativa positiva
0.40625 0.59375
Group means:
      inibina$difinib
negativa      49.01385
positiva      202.70158
Coefficients of linear discriminants:
              LD1
inibina$difinib 0.007050054
```

A função considera as proporções de casos negativos (41%) e positivos (59%) no conjunto de dados de treinamento como probabilidades *a priori*, dado que elas não foram especificadas no comando.

O coeficiente da função discriminante (0.00705) corresponde à combinação linear de difinib usada para a decisão difere daquele obtido acima (0,0076) pois a função lda() considera uma transformação com a finalidade de deixar os resultados com variância unitária (o que não influi na classificação).

## Classificador de Bayes: exemplo 2

```
lda(inibina$resposta ~ inibina$difinib, data = inibina)
Prior probabilities of groups:
negativa positiva
0.40625 0.59375
Group means:
      inibina$difinib
negativa      49.01385
positiva      202.70158
Coefficients of linear discriminants:
              LD1
inibina$difinib 0.007050054
```

A função considera as proporções de casos negativos (41%) e positivos (59%) no conjunto de dados de treinamento como probabilidades *a priori*, dado que elas não foram especificadas no comando.

O coeficiente da função discriminante (0.00705) corresponde à combinação linear de difinib usada para a decisão difere daquele obtido acima (0,0076) pois a função lda() considera uma transformação com a finalidade de deixar os resultados com variância unitária (o que não influi na classificação).

## Classificador de Bayes: exemplo 2

Uma tabela relacionando a classificação predita com os valores reais da resposta pode ser obtido por meio dos comandos

```
predito <- predict(fisher)
table(predito$class, inibina$resposta)
    negativa positiva
negativa      9      2
positiva      4     17
```

indicando que a probabilidade de classificação correta é 81%, ligeiramente superior ao que foi conseguido com o emprego de regressão logística (ver Exemplo 6.9). Histogramas para os valores da função discriminante calculada para cada elemento do conjunto de dados estão dispostos na Figura 1.

## Classificador de Bayes: exemplo 2

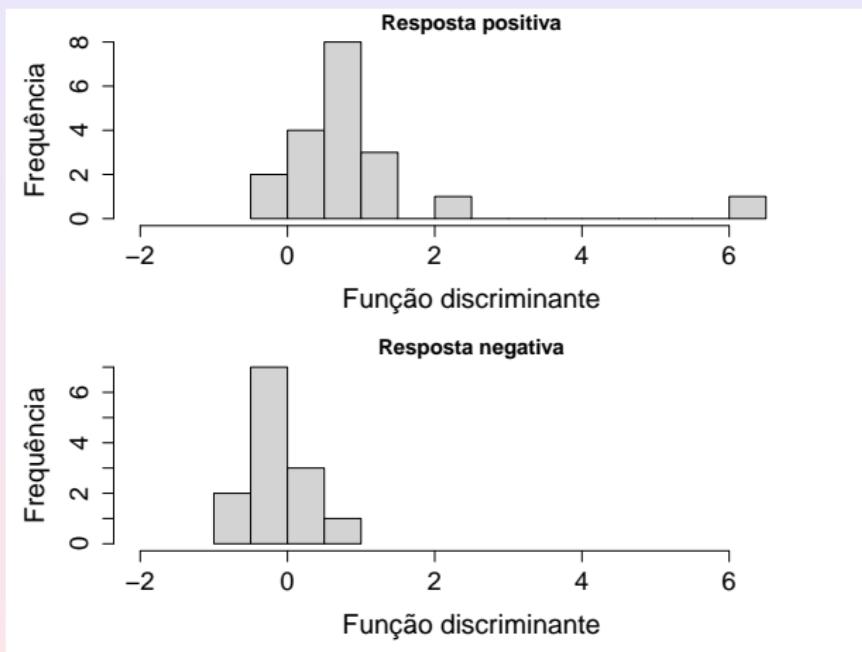


Figura 1: Histogramas para valores da função discriminante.

## Função discriminante de Fisher

- Consideremos novamente o caso de duas classes (ou populações),  $\mathcal{G}_1$  e  $\mathcal{G}_2$  para as quais pretendemos obter um classificador com base em um vetor de variáveis preditoras,  $\mathbf{X} = (X_1, \dots, X_p)^\top$ .
- A ideia de Fisher é considerar uma combinação linear  $Y = \ell^\top \mathbf{X}$ , com  $\ell = (\ell_1, \dots, \ell_p)^\top$  de modo que o conjunto de variáveis preditoras seja transformado numa variável escalar  $Y$ .
- Sejam  $\mu_{1Y}$  e  $\mu_{2Y}$ , respectivamente, as médias de  $Y$  obtidas dos valores de  $\mathbf{X}$  associadas aos dados  $\mathcal{G}_1$  e  $\mathcal{G}_2$ . A regra para classificação consiste em selecionar a combinação linear que maximiza a distância quadrática entre essas duas médias, relativamente à variabilidade dos valores de  $Y$ .
- Uma suposição adicional e, às vezes, irrealista, é que as matrizes de covariâncias

$$\boldsymbol{\Sigma}_i = E(\mathbf{X} - \boldsymbol{\mu}_i)(\mathbf{X} - \boldsymbol{\mu}_i)^\top, \quad (16)$$

$i = 1, 2$ , em que  $\boldsymbol{\mu}_1 = E(\mathbf{X}|\mathcal{G}_1)$  e  $\boldsymbol{\mu}_2 = E(\mathbf{X}|\mathcal{G}_2)$ , sejam iguais para as duas classes, isto é,  $\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2 = \boldsymbol{\Sigma}$ .

## FDLF

- Consequentemente,

$$\sigma_Y^2 = \text{var}(\ell^\top \mathbf{X}) = \ell^\top \boldsymbol{\Sigma} \ell$$

é igual para ambas as classes.

- 

$$\mu_{1Y} = E(Y|\mathcal{G}_1) = \ell^\top \boldsymbol{\mu}_1 \quad \text{e} \quad \mu_{2Y} = E(Y|\mathcal{G}_2) = \ell^\top \boldsymbol{\mu}_2$$

e a razão

$$\begin{aligned} \frac{(\mu_{1Y} - \mu_{2Y})^2}{\sigma_Y^2} &= \frac{(\ell^\top \boldsymbol{\mu}_1 - \ell^\top \boldsymbol{\mu}_2)^2}{\ell^\top \boldsymbol{\Sigma} \ell} = \frac{\ell^\top (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^\top \ell}{\ell^\top \boldsymbol{\Sigma} \ell} \\ &= \frac{(\ell^\top \boldsymbol{\delta})^2}{\ell^\top \boldsymbol{\Sigma} \ell}, \end{aligned} \tag{17}$$

com  $\boldsymbol{\delta} = \boldsymbol{\mu}_1 - \boldsymbol{\mu}_2$  é maximizada se

$$\ell = c \boldsymbol{\Sigma}^{-1} \boldsymbol{\delta} = c \boldsymbol{\Sigma}^{-1} (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2) \tag{18}$$

para todo  $c \neq 0$ .

- No caso  $c = 1$ , obtemos a função discriminante linear de Fisher

$$Y = \ell^\top \mathbf{X} = (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^\top \boldsymbol{\Sigma}^{-1} \mathbf{X}. \tag{19}$$

e o valor máximo da razão (17) é  $\boldsymbol{\delta}^\top \boldsymbol{\Sigma}^{-1} \boldsymbol{\delta}$ .

## FDLF

- Para uma nova observação  $\mathbf{x}_0$ , sejam  $y_0 = (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^\top \boldsymbol{\Sigma}^{-1} \mathbf{x}_0$  e

$$\mu = \frac{\mu_{1Y} + \mu_{2Y}}{2} = \frac{1}{2}(\ell^\top \boldsymbol{\mu}_1 + \ell^\top \boldsymbol{\mu}_2) \quad (20)$$

(o ponto médio entre as médias univariadas associadas às duas classes).  
Em virtude de (19), esse ponto médio pode ser expresso como

$$\mu = \frac{(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^\top \boldsymbol{\Sigma}^{-1} (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)}{2}, \quad (21)$$

- Consequentemente,

$$E(Y_0|\mathcal{G}_1) - \mu \geq 0 \text{ e } E(Y_0|\mathcal{G}_2) - \mu < 0.$$

e uma **regra de classificação** é

Classifique  $\mathbf{x}_0$  em  $\mathcal{G}_1$  se  $y_0 \geq \mu$ ,  
Classifique  $\mathbf{x}_0$  em  $\mathcal{G}_2$  se  $y_0 < \mu$ .

## Estimativa da FDLF

- Normalmente,  $\mu_1$ ,  $\mu_2$  e  $\Sigma$  são desconhecidas e têm que ser estimadas a partir de amostras de  $\mathcal{G}_1$  e  $\mathcal{G}_2$ , denotadas por  $\mathbf{X}_1 = [\mathbf{x}_{11}, \dots, \mathbf{x}_{1,n_1}]$ , uma matriz com dimensão  $p \times n_1$  e  $\mathbf{X}_2 = [\mathbf{x}_{21}, \dots, \mathbf{x}_{2,n_2}]$ , uma matriz com dimensão  $p \times n_2$ .
- Com os dados dessas amostras, podemos obter estimativas  $\bar{\mathbf{x}}_1, \bar{\mathbf{x}}_2, \mathbf{S}_1$  e  $\mathbf{S}_2$ , das médias e da matriz de covariâncias comum  $\Sigma$ , para a qual um estimador não enviesado é

$$\mathbf{S}_p = \frac{(n_1 - 1)\mathbf{S}_1 + (n_2 - 1)\mathbf{S}_2}{n_1 + n_2 - 2}. \quad (22)$$

- A função discriminante estimada é  $\hat{\ell}^\top \mathbf{x} = (\bar{\mathbf{x}}_1 - \bar{\mathbf{x}}_2)^\top \mathbf{S}_p \mathbf{x}$  e a regra de classificação é:

Classifique a observação  $\mathbf{x}_0$  em  $\mathcal{G}_1$  se  $y_0 - \hat{\mu} \geq 0$ ,

Classifique a observação  $\mathbf{x}_0$  em  $\mathcal{G}_2$  se  $y_0 - \hat{\mu} < 0$ ,

em que  $\hat{\mu} = (\bar{\mathbf{x}}_1 - \bar{\mathbf{x}}_2)^\top \mathbf{S}_p^{-1} (\bar{\mathbf{x}}_1 + \bar{\mathbf{x}}_2)$ .

- Nas demais expressões, os parâmetros são substituídas pelas respectivas estimativas.
- Outra suposição comumente adotada é que as variáveis preditoras têm distribuição Normal multivariada. Nesse caso, a solução encontrada por meio da função discriminante linear de Fisher é ótima.

## Classificador KNN

- Vimos que o **classificador de Bayes** associa cada observação de teste com o valor do preditor  $x_0$  à classe  $j$  de forma que

$$P(Y = j|X = x_0) \quad (23)$$

seja a maior possível.

- No caso de duas classes, a observação será associada à Classe 1 se  $P(Y = 1|X = x_0) > 0,5$  e à Classe 2, se  $P(Y = 0|X = x_0) < 0,5$ . A **fronteira de Bayes** é  $P(Y = 1|X = x_0) = 0,5$ .
- A **taxa de erro de Bayes global** é  $1 - E(\max_j P(Y = j|X))$ , obtida com base na média de todas as taxas de erro sobre todos os valores possíveis de  $j$ .
- Na prática como não conhecemos a distribuição condicional de  $Y$ , dado  $X$ , precisamos estimar essa probabilidade condicional, o que pode ser efetivado por meio de um método conhecido por ***K*-ésimo vizinho mais próximo (*K*-nearest neighbor, KNN)**.

# Classificador KNN

O algoritmo associado a esse método é:

- i) Fixe  $K$  e uma observação teste  $x_0$ ;
- ii) Identifique  $K$  pontos do conjunto de dados de treinamento que sejam os mais próximos de  $x_0$  segundo alguma medida de distância; denote esse conjunto por  $\mathcal{V}_0$ ;
- iii) Estime a probabilidade condicional de que a observação teste pertença à Classe  $j$  como a fração dos pontos de  $\mathcal{V}_0$  cujos valores de  $Y$  sejam iguais a  $j$ , ou seja, como

$$P(Y = j | X = x_0) = \frac{1}{K} \sum_{i \in \mathcal{V}_0} I(y_i = j). \quad (24)$$

- iv) classifique  $x_0$  na classe associada à maior probabilidade.

A função `knn()` do pacote `caret` pode ser utilizada com essa finalidade.

## Classificador KNN-Exemplo

- **Exemplo.** Consideremos, novamente, os dados do arquivo **inibina** utilizando a variável **difinib** como preditora e adotemos a estratégia de validação cruzada por meio do método LOOCV. Além disso, avaliemos o efeito de considerar entre 1 e 5 vizinhos mais próximos no processo de classificação.
- Os comandos necessários para a concretização da análise são

```
set.seed(2327854)
trControl <- trainControl(method = "LOOCV")

fit <- train(resposta ~ difinib, method = "knn",
             tuneGrid = expand.grid(k = 1:5),
             trControl = trControl, metric= "Accuracy",
             data = inibina)

fit
```

## Classificador KNN-Exemplo

Os resultados correspondentes são:

k-Nearest Neighbors

32 samples

1 predictor

2 classes: 'negativa', 'positiva'

No pre-processing

Resampling: Leave-One-Out Cross-Validation

Summary of sample sizes: 31, 31, 31, 31, 31, 31, ...

Resampling results across tuning parameters:

k	Accuracy	Kappa
1	0.71875	0.4240000
2	0.78125	0.5409836
3	0.81250	0.6016598
4	0.78125	0.5409836
5	0.81250	0.6016598

Accuracy was used to select the optimal model using the largest value.

The final value used for the model was k = 5.

## Classificador KNN-Exemplo

- Segundo o esse processo, o melhor resultado (com K=5 vizinhos) gera uma acurácia (média) de 81.3%. A tabela de classificação obtida por meio do ajuste do modelo final ao conjunto de dados original, juntamente com estatísticas descritivas pode ser obtida por meio dos comandos:

```
predito <- predict(fit)  
confusionMatrix(predito, inibina$resposta)
```

- que geram os seguintes resultados:

```
Confusion Matrix and Statistics  
Reference
```

```
Prediction negativa positiva
```

negativa	9	1
positiva	4	18

```
Accuracy : 0.8438  
95% CI : (0.6721, 0.9472)
```

```
No Information Rate : 0.5938
```

```
P-Value [Acc > NIR] : 0.002273
```

```
Kappa : 0.6639
```

```
Mcnemar's Test P-Value : 0.371093
```

## Classificador KNN-Exemplo

```
Sensitivity : 0.6923
Specificity : 0.9474
Pos Pred Value : 0.9000
Neg Pred Value : 0.8182
Prevalence : 0.4062
Detection Rate : 0.2812
Detection Prevalence : 0.3125
Balanced Accuracy : 0.8198
```

A acurácia é de 84,4%, sensibilidade de 69,2% e especificidade de 94,7%.

## Algumas medidas

Considere a tabela (matriz de confusão):

		Condição Verdadeira	
		Positiva	Negativa
Condição Prevista	Positiva	$n_{11}$ ( <b>TP</b> )	$n_{12}$ ( <b>FP</b> )
	Negativa	$n_{21}$ ( <b>FN</b> )	$n_{22}$ ( <b>TN</b> )
		<b>CP</b>	<b>CN</b>

TP=True positive, FP= False positive, FN= False negative, TN= True negative

$$\text{TPR} = \frac{TP}{TP+FN} = \frac{n_{11}}{n_{11}+n_{21}} \text{ (sensibilidade) (True positive rate)}$$

$$\text{TNR} = \frac{TN}{FP+TN} = \frac{n_{22}}{n_{12}+n_{22}} \text{ (especificidade) (True negative rate)}$$

$$\text{Prevalence} = \frac{CP}{CP+CN}$$

$$\text{PPV} = \frac{TP}{TP+FP} \text{ (Positive predictive value, precision)}$$

$$\text{NPV} = \frac{TN}{FN+TN} \text{ (Negative predictive value)}$$

$$\text{FDR} = \frac{FP}{TP+FP} \text{ (False discovery rate)}$$

$$\text{Accuracy} = \text{ACC} = \frac{TP+TN}{CP+CN}, \text{ Balanced accuracy} = \text{BA} = \frac{\text{TPR}+\text{TNR}}{2}$$

## Teste de McNemar

- É um teste para dados nominais pareados, dispostos numa tabela de contingência  $2 \times 2$  (McNemar, 1947).
- Considere a tabela, com resultados de dois testes para  $n$  indivíduos:

		Teste 2	
		Positivo	Negativo
Teste 1	Positivo	$n_{11}$	$n_{12}$
	Negativo	$n_{21}$	$n_{22}$
		$n_{\cdot 1}$	$n_{\cdot 2}$
			1

- Sejam  $p_{ij} = n_{ij}/n$ ,  $i, j = 1, 2$ ,  $p_{\cdot i}$  as soma das linhas,  $i = 1, 2$   $p_{\cdot j}$  as somas das colunas,  $j = 1, 2$  (com os  $p_{ij}$  substituindo os  $n_{ij}$  na tabela). A hipótese nula de homogeneidade marginal afirma que as duas probabilidades marginais de cada resultado são iguais, isto é,  $p_{11} + p_{12} = p_{11} + p_{21}$  e  $p_{21} + p_{22} = p_{12} + p_{22}$ , ou seja,  $p_{1\cdot} = p_{\cdot 1}$  e  $p_{2\cdot} = p_{\cdot 2}$ , ou ainda

$$H_0 : p_{12} = p_{21},$$

$$H_1 : p_{12} \neq p_{21}.$$

- A estatística de MacNemar é

$$M = \frac{(n_{12} - n_{21})^2}{n_{12} + n_{21}},$$

que, sob  $H_0$ , tem uma distribuição qui-quadrado com 1 grau de liberdade.

## Teste de McNemar

- Se  $n_{12}$  ou  $n_{21}$  for pequeno ( soma < 25), então  $M$  não é bem aproximada pela distribuição qui-quadrado. Um teste binomial exato pode ser usado para  $n_{21}$ . Para  $n_{12} > n_{21}$ , o valor- $p$  exato é

$$p = 2 \sum_{i=n_{12}}^N \binom{N}{i} (1/2)^i (1/2)^{N-i},$$

com  $N = n_{12} + n_{21}$ .

- Edwards (1948) propôs a seguinte correção de continuidade para  $M$ :

$$M = \frac{(|n_{12} - n_{21}| - 1)^2}{n_{12} + n_{21}}.$$

- No exemplo,  $M = (|1 - 4| - 1)^2 / 5 = 0,8$  e valor- $p$  é  $P(\chi_1^2 > 0,8 | H_0) \approx 0,37$ , logo não rejeitamos  $H_0$ .

## Aplicação do teste em ML

- O teste de McNemar pode ser usado para comparar técnicas de classificação, para aqueles algoritmos que são usados em conjuntos de dados grandes e não podem ser repetidos via algum método de reamostragem, como CV.
- Dietterich (1998) considerou 5 testes para determinar se um algoritmo de classificação é melhor do que um outro, em um particular conjunto de dados. O objetivo era determinar a probabilidade de erro de tipo I de cada teste.
- Testes que não devem ser usados:(a) teste para a diferença de duas proporções; (b) teste  $t$  pareado (diferenças) baseado em partições aleatórias dos conjuntos de treinamento/teste; (c) teste  $t$  pareado baseado em 10-fold CV. Todos exibem probabilidades de erro de tipo I altas.
- O teste de McNemar tem baixa probabilidade de erro do tipo I.
- O autor introduziu um teste,  $5 \times 2$  CV, baseado em 5 iterações de uma 2-fold CV, que tem uma probabilidade de erro de tipo I aceitável e poder maior do que o teste de McNemar.

## Aplicação do teste em ML

- Questão: Dados dois classificadores  $C_1$  e  $C_2$  e dados suficientes para aplicá-los em um conjunto de teste, determinar qual classificador será mais acurado em novos conjuntos de testes.
- Essa questão pode ser respondida medindo-se a acurácia de cada classificador no conjunto teste a aplicando o teste de McNemar.
- Dietterich (1998) considera 9 questões, algumas ainda não respondidas, e foca seu artigo na seguinte questão: **Dados dois algoritmos de aprendizagem A e B, e um conjuntos de dados pequeno S, qual algoritmo produzirá classificadores mais acurados quando treinados em conjuntos de dados do mesmo tamanho que S?**
- Para isso, é necessário usar métodos de reamostragem. Ele compara vários testes estatísticos para responder a questão.
- O primeiro passo é identificar as fontes de variação que podem ser controladas por cada teste.
- **4 fontes de variação:** (a) variação aleatória na seleção do conjunto de teste que será usado para avaliar os algoritmos; (b) variação na escolha dos dados de treinamento (instabilidade); (c) aleatoriedade interna do algoritmo de aprendizagem. Por exemplo, o algoritmo **backpropagation** depende dos pesos (aleatórios) iniciais; (d) erro de classificação aleatório.

## Aplicação do teste em ML

- Um teste deve concluir que dois algoritmos são diferentes se, e somente se, suas taxas de classificação corretas sejam diferentes, em média, quando treinados em um conjunto de treinamento de tamanho fixo e testado em todos os dados da população.
- Para tanto, o teste deve considerar o tamanho do teste (probabilidade do erro de tipo I) e executar o algoritmo múltiplas vezes e medir a variação da acurácia dos classificadores resultantes
- Para aplicar o teste de McNemar, dividimos a amostra de dados S em um conjunto de treinamento  $T_0$  (com  $n$  observações) e um conjunto teste  $T_1$  (com  $m$  observações). Treinamos os algoritmos  $C_1$  e  $C_2$  no conjunto  $T_0$ , obtendo-se classificadores  $\hat{C}_1$  e  $\hat{C}_2$ . Então, testamos esses classificadores no conjunto  $T_1$ . Para cada  $x \in T_1$ , registramos como esse ponto foi classificado e construímos a seguinte tabela  $2 \times 2$ :

		Classificador 2	
		Class. Correta	Class. Errônea
Classificador 1	Class. correta	$n_{11} = \text{Sim/Sim}$	$n_{12} = \text{Sim/Não}$
	Class. errônea	$n_{21} = \text{Não/Sim}$	$n_{22} = \text{Não/Não}$

- $\sum_i \sum_j n_{ij} = m$ .

## Aplicação do teste em ML

- Rejeitando  $H_0$ , os dois algoritmos terão desempenho diferentes quando treinados em  $T_0$ .
- Note que esse teste tem dois problemas: primeiro, não mede diretamente a variabilidade devida à escolha de  $T_0$ , nem a aleatoriedade interna do algoritmo, pois um único conjunto de treinamento é escolhido. Segundo, ele não compara os desempenhos dos algoritmos em conjuntos de treinamento de tamanho  $|S|$ , mas sobre conjuntos de tamanho  $n$ , que deve ser menor do que  $|S|$ , para que tenhamos um conjunto de teste grande.

## AD quadrática

- O classificador obtido por meio de Análise Discriminante Quadrática supõe, como na Análise Discriminante Linear usual, que as observações são extraídas de uma distribuição gaussiana multivariada, mas não necessita da suposição de homocedasticidade (matrizes de covariâncias iguais), ou seja, admite que a cada classe esteja associada uma matriz de covariância,  $\Sigma_k$ .
- Como no caso da Análise Discriminante Linear, não é difícil ver que o classificador de Bayes associa um elemento com valor das variáveis preditora igual a  $x$  à classe para a qual a função quadrática

$$\begin{aligned}\delta_k(x) &= -\frac{1}{2}(x - \mu_k)^\top \Sigma_k^{-1} (x - \mu_k) - \frac{1}{2} \log |\Sigma_k| + \log \pi_k \\ &= -\frac{1}{2}x^\top \Sigma_k^{-1} x + x^\top \Sigma_k^{-1} \mu_k - \frac{1}{2} \mu_k^\top \Sigma_k^{-1} \mu_k - \frac{1}{2} \log |\Sigma_k| + \log \pi_k.\end{aligned}\tag{25}$$

é máxima.

## AD quadrática

- Como os elementos de (25) não são conhecidos, pode-se estimá-los por meio de

$$\bar{\mathbf{x}}_k = \frac{1}{n_k} \sum_{i=1}^{n_k} \mathbf{x}_i, \quad \mathbf{S}_k = \frac{1}{n_k} \sum_{i=1}^{n_k} (\mathbf{x}_i - \bar{\mathbf{x}}_k)(\mathbf{x}_i - \bar{\mathbf{x}}_k)^{\top}$$

e  $\hat{\pi}_k = n_k/n$ , em que  $n_k$  é o número de observações na classe  $k$  e  $n$  é o número total de observações no conjunto de dados. O primeiro termo do segundo membro da primeira igualdade de (25) é a **distância de Mahalanobis**.

- O número de parâmetros a estimar,  $Kp(p+1)/2$ , é maior que no caso de Análise Discriminante Linear, na qual a matriz de covariâncias é comum. Além disso, a versão linear apresenta variância substancialmente menor mas viés maior do que a versão quadrática. A Análise Discriminante Quadrática é recomendada se o número de dados for grande; em caso contrário, convém usar Análise Discriminante Linear.

## AD regularizada

- A Análise Discriminante Regularizada foi proposta por Friedman (1989) e é um compromisso entre Análise Discriminante Linear e Análise Discriminante Quadrática.
- O método proposto por Friedman consiste em “encolher” (*shrink*) as matrizes de covariâncias da Análise Discriminante Quadrática em direção a uma matriz de covariâncias comum.
- Friedman (1989) propõe o seguinte procedimento de regularização

$$\boldsymbol{\Sigma}_k(\lambda, \gamma) = (1 - \gamma)\boldsymbol{\Sigma}_k(\lambda) + \frac{\gamma}{p} \text{tr}[\boldsymbol{\Sigma}_k(\lambda)]\mathbf{I}, \quad (26)$$

em que  $\text{tr}(\mathbf{A})$  indica o traço da matriz  $\mathbf{A}$ ,  $\mathbf{I}$  é a matriz identidade e

$$\boldsymbol{\Sigma}_k(\lambda) = \lambda\boldsymbol{\Sigma}_k + (1 - \lambda)\boldsymbol{\Sigma} \quad (27)$$

com  $\boldsymbol{\Sigma} = \sum n_k \boldsymbol{\Sigma}_k / n$ .

## AD regularizada

- O parâmetro  $\lambda \in [0, 1]$  controla o grau segundo o qual a matriz de covariâncias ponderada pode ser usada e  $\gamma \in [0, 1]$  controla o grau de encolhimento ao autovalor médio. Na prática,  $\lambda$  e  $\gamma$  são escolhidos por meio de LOOVC para cada ponto de uma grade no quadrado unitário.
- Quando  $\lambda = 1$  e  $\gamma = 0$ , a Análise Discriminante Regularizada reduz-se à Análise Discriminante Linear. Se  $\lambda = 0$  e  $\gamma = 0$ , o método reduz-se à Análise Discriminante Quadrática. Se  $p > n_k$ , para todo  $k$  e  $p < n$ ,  $\Sigma_k$  é singular, mas nem a matriz ponderada  $\Sigma$  nem  $\Sigma_k(\lambda)$  em (27) o são. Se  $p > n$ , todas essas matrizes são singulares e a matriz  $\Sigma_k(\lambda, \gamma)$  é regularizada por (26).
- Essas análises podem ser concretizadas por meio das funções `qda()` do pacote MASS e `rda()` do pacote klaR.

## ADQ e ADR - exemplo

Vamos considerar novamente o conjunto de dados disco do Exemplo 8.1. Na segunda análise realizada por meio de regressão logística, a classificação foi concretizada via validação cruzada VC5/5 e LOOCV tendo como variáveis preditoras a distância aberta, a distância fechada ou ambas.

A melhor acurácia, 85,7% foi obtida com ambas as variáveis preditoras e VC5/5.

Agora, consideramos a classificação realizada por intermédio de Análises Discriminantes Linear, Quadrática e Regularizada, separando os dados em um conjunto de treinamento contendo 80% (83) dos elementos, selecionados aleatoriamente, e em um conjunto de validação com os restantes 21 elementos. Lembremos que  $y = 1$  corresponde a discos deslocados e  $y = 0$  a discos não deslocados.

As acurárias obtidas por meio de Análise Discriminante Linear e Análise Discriminante Quadrática foram, respectivamente, 90% e 85%.

## Referências

- Dietterich, T. G. (1998). Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Computation*, **10**, 1895–1923.
- Edwards, A (1948). Note on the correction for continuity in testing the significance of the difference between correlated proportions. *Psychometrika*, **13**, 185–187.
- James, G., Witten, D., Hastie, T. and Tibshirani, R. (2017). *Introduction to Statistical Learning*. Springer.
- McNemar, Q. (1947). Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, **12**, 153–157.
- Morettin, P. A. e Singer, J. M. (2022). *Estatística e Ciência de Dados*. LTC: Rio de Janeiro.

# MAE 5905: Introdução à Ciência de Dados

Pedro A. Morettin

Instituto de Matemática e Estatística  
Universidade de São Paulo  
[pam@ime.usp.br](mailto:pam@ime.usp.br)  
<http://www.ime.usp.br/~pam>

## Aula 9

5 de maio de 2023

# Sumário

1 Algoritmos de Suporte Vetorial

2 Classificador de Margem Máxima

3 Classificador de Margem Flexível

# ASV/SVM

- Algoritmos de Suporte Vetorial (ASV), conhecidos na literatura anglo-saxônica como **Support Vector Machines** (SVM) foram introduzidos por Cortes and Vapnik (1995), que desenvolveram essa classe de algoritmos para classificação binária e englobam técnicas úteis para classificação, com inúmeras aplicações, dentre as quais destacamos reconhecimento de padrões, classificação de imagens, reconhecimentos de textos escritos à mão, expressão de gens em DNAs etc.
- Vapnik and Chervonenkis (1964, 1974) foram, talvez, os primeiros a usar o termo **Aprendizado com Estatística** (**Statistical Learning**) em conexão com problemas de reconhecimento de padrões e inteligência artificial.
- Algoritmos de suporte vetorial são generalizações não lineares do algoritmo *Generalized Portrait*, desenvolvido por Vapnik e Chervonenkis (1964).
- Embora a tradução literal do termo proposto por Vapnik seja **Máquinas** de Suporte Vetorial, optamos por utilizar **Algoritmos** de Suporte Vetorial para que não se pense que algum tipo de máquina esteja ligado a essas técnicas. Aparentemente, Vapnik utilizou esse termo para enfatizar o aspecto computacional intrínseco à aplicação dos algoritmos.

## ASV/SVM

- Uma propriedade importante dos ASV é que a determinação dos parâmetros do modelo corresponde a um problema de otimização convexa, de modo que qualquer solução local é também uma solução global.
- Fazem uso extensivo de Multiplicadores de Lagrange.
- Os algoritmos de suporte vetorial competem com outras técnicas bastante utilizadas, como Modelos Lineares Generalizados (MLG), Modelos Aditivos Generalizados (MAG), Redes Neuronais (Neurais), modelos baseados em árvores etc.
- A comparação com esses métodos é baseada em três fatores: **interpretabilidade** do modelo usado, **desempenho** na presença de valores atípicos e **poder preditivo**.
- Por exemplo, os MLG têm baixo desempenho na presença de valores atípicos, valor preditivo moderado e boa interpretabilidade.
- Por outro lado, os ASV têm desempenho moderado na presença de valores atípicos, alto poder preditivo e baixa interpretabilidade.
- ASV são usados para AE Supervisionado (regressão e classificação).

# ASV/SVM

- O princípio operacional fundamental dos ASV é que um **kernel** (núcleo) é usado para mapear os dados de entrada (ou padrões) em um espaço de dimensão mais alta (**feature space**), de tal sorte que o problema de classificação, por exemplo, torna-se separável.
- O sucesso da aplicação dos ASV depende da escolha, a priori, desse kernel.
- Os kernels mais populares são:

Gaussiano

Polinomial

Exponential radial basis

Splines

- Recentemente, têm sido usados kernels baseados em ondaletas.
- Essencialmente, um ASV é implementado por um código computacional que realiza essas tarefas. No Repositório R há pacotes como **e1071** e a função **svm** desenvolvidos com essa finalidade. Outras alternativas são o pacote **kernlab** e a função **ksvm**.

# ASV/SVM

A abordagem de Cortes and Vapnik (1995) para o problema de classificação baseia-se nas seguintes premissas:

- a) **Separação de classes**: procura-se o melhor hiperplano separador entre as classes, maximizando-se a **margem** entre os pontos mais próximos das duas classes. Os pontos sobre as fronteiras dessas classes são chamados **vetores suporte (support vectors)**.
- b) **Superposição de classes**: pontos de uma classe que estão no outro lado do hiperplano separador são ponderados com baixo peso para reduzir sua influência.
- c) **Não linearidade**: quando não pudermos encontrar um separador linear, utilizamos um **kernel** para mapear os dados de entrada em um espaço de dimensão mais alta (**feature space**) de tal forma que nesse espaço, são construídos os hiperplanos separadores.
- d) **Solução do problema**: o problema envolve otimização quadrática e pode ser resolvido com técnicas conhecidas.

## Fundamentação dos ASV

- Apresentaremos as ideias básicas sobre algoritmos de suporte vetorial (ASV), concentrando-nos no problema de **classificação dicotômica**, i.e., em que as unidades amostrais devem ser classificadas em uma de duas classes possíveis. Para ideias sobre o caso de mais de duas classes, veja o Texto.
- Adotaremos uma abordagem heurística, mais próxima daquela usualmente empregada em Estatística, deixando para as Notas de Capítulo do Texto a abordagem original (e mais formal) dos ASV.
- Seja  $\mathcal{X}$  o **espaço dos dados** (ou dos padrões); em geral,  $\mathcal{X} = \mathbb{R}^d$  e seja a resposta  $y \in \{-1, 1\}$ .
- Por exemplo, podemos ter dados de várias variáveis explicativas (idade, peso, taxa de colesterol etc.) e uma variável resposta (doença cardíaca, com  $y = 1$  em caso afirmativo e  $y = -1$  em caso negativo) observadas em vários indivíduos (o **conjunto de treinamento**). O problema de classificação consiste na determinação de dois subconjuntos (classes) de  $\mathcal{X}$ , um das quais estará associado a indivíduos com doença cardíaca. O classificador indicará em qual das classes deveremos incluir novos indivíduos (**o conjunto de teste**) para os quais conhecemos os valores das variáveis explicativas.

## Fundamentação dos ASV

Vamos considerar três situações:

- 1) As classes são perfeitamente separáveis por uma fronteira linear; nesse caso, o separador (hiperplano) é conhecido como **classificador de margem máxima** (CMM).
  - Para duas variáveis, o separador é uma reta; para três variáveis, o separador é um plano. No caso de  $p$  variáveis, o separador é um **hiperplano** de dimensão  $p - 1$ . A Figura 1 é um exemplo. Note que podemos ter mais de uma reta separando as duas classes.
- 2) Não há um hiperplano que separe as duas classes, como no exemplo apresentado na Figura 2, que corresponde à Figura 2 com pontos trocados de lugar. O separador, neste caso é o **classificador de margem flexível** (CMF).
- 3) Um separador linear não conduz a resultados satisfatórios exigindo a definição de fronteiras de separação não lineares. Para isso, recorremos ou a funções não lineares das observações ou a **kernels**, para mapear o espaço dos dados em um espaço de dimensão maior. O separador, neste caso é o **classificador de margem não linear** (CMNL).

## Fundamentação dos ASV

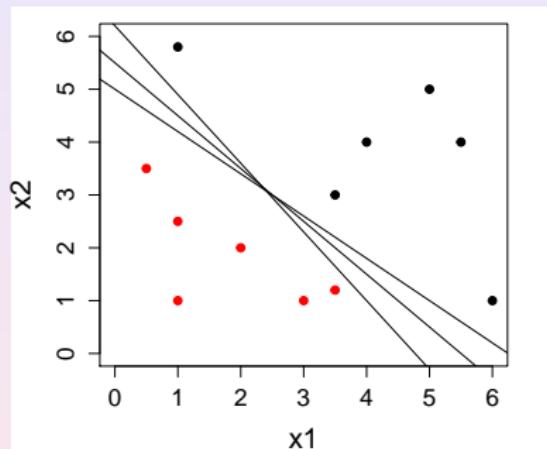


Figura 1: Dois conjuntos de pontos perfeitamente separáveis por um hiperplano (reta).

## Fundamentação dos ASV

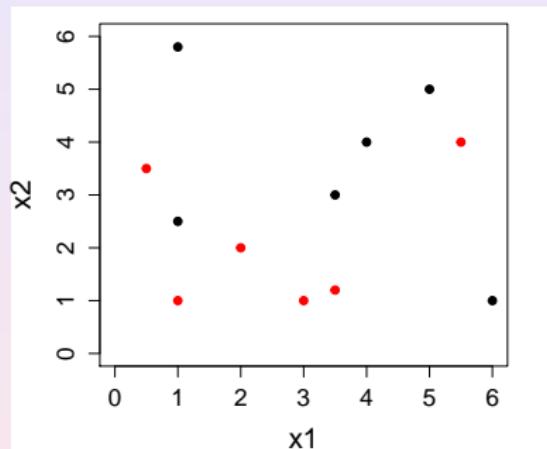


Figura 2: Dois conjuntos de pontos não separáveis por um hiperplano (reta).

## Margem e Vetores Suporte

- No caso de duas variáveis, o hiperplano é uma reta com equação  $\alpha + \beta_1 X_1 + \beta_2 X_2 = 0$ .
- Essa reta separa o plano em duas regiões, uma em que  $\alpha + \beta_1 X_1 + \beta_2 X_2 > 0$  e outra em que  $\alpha + \beta_1 X_1 + \beta_2 X_2 < 0$ .
- Consideremos  $n$  observações das variáveis  $X_1, \dots, X_p$ , dispostas na forma de uma matriz  $\mathbf{X}$ , de ordem  $n \times p$ . Seja  $\mathbf{x}_i = (x_{i1}, \dots, x_{ip})^\top$ , o vetor correspondente à  $i$ -ésima coluna de  $\mathbf{X}$ .
- Além disso, sejam  $y_1, \dots, y_n \in \{-1, 1\}$ , definindo o conjunto de treinamento  $\mathcal{T} = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)\}$  e seja  $\mathbf{x}_0 = (x_{10}, \dots, x_{p0})^\top$  um **vetor de teste**.

## Margem e Vetores Suporte

- Queremos desenvolver um classificador usando um hiperplano separador no espaço  $\mathbb{R}^p$  com base no conjunto de treinamento.
- Definindo  $\beta = (\beta_1, \dots, \beta_p)^\top$ , teremos

$$\alpha + \beta^\top \mathbf{x}_i > 0, \quad \text{se } y_i = 1, \quad (1)$$

$$\alpha + \beta^\top \mathbf{x}_i < 0, \quad \text{se } y_i = -1. \quad (2)$$

- Chamemos

$$f(\mathbf{x}) = \alpha + \beta^\top \mathbf{x}. \quad (3)$$

Então, classificaremos  $\mathbf{x}_0$  a partir do sinal de  $f(\mathbf{x}_0) = \alpha + \beta^\top \mathbf{x}_0$ ; se o sinal for positivo,  $\mathbf{x}_0$  será classificado na Classe 1 (para a qual  $y = 1$ , digamos), e se o sinal for negativo, na Classe 2 (para a qual  $y = -1$ ). Em qualquer situação,  $y_i(\alpha + \beta^\top \mathbf{x}_i) \geq 0$ .

## Margem e Vetores Suporte

- Como vimos, podem existir infinitos hiperplanos separadores, se os dados de treinamento estiverem perfeitamente separados.
- A sugestão de Vapnik e colaboradores é escolher um hiperplano que esteja o mais afastado das observações de treinamento, chamado de **hiperplano de margem máxima**.
- A **margem** é a menor distância entre o hiperplano e os pontos de treinamento.
- O classificador de margem máxima (CMM) é a solução (se existir) do seguinte problema de otimização:

$$\text{maximizar}_{(\alpha, \beta)} m(\alpha, \beta) \quad (4)$$

sujeito a

$$\sum_{i=1}^p \beta_i^2 = 1, \quad (5)$$

$$y_i(\alpha + \beta^\top \mathbf{x}_i) \geq m(\alpha, \beta), \quad i = 1, \dots, n. \quad (6)$$

- Dizemos que  $m = m(\alpha, \beta)$  é a **margem** do hiperplano e cada observação estará do lado correto do hiperplano se  $m > 0$ .

## Margem e Vetores Suporte

- Os chamados **vetores suporte** são definidos pelos pontos cujas distâncias ao hiperplano separador sejam iguais à margem e se situam sobre as **fronteiras de separação**, que são hiperplanos "paralelos" cujas distâncias ao hiperplano separador é igual à margem.
- O classificador depende desses vetores, mas não das demais observações.
- A distância  $m$  do hiperplano separador a um ponto do conjunto de treinamento é

$$m = |f(\mathbf{x})|/||\boldsymbol{\beta}||,$$

em que o denominador indica a norma do vetor  $\boldsymbol{\beta}$ .

## Margem e Vetores Suporte

- Como o interesse está nos pontos que são corretamente classificados, devemos ter  $y_i f(\mathbf{x}_i) > 0$ ,  $i = 1, \dots, n$ . Então

$$\frac{y_i f(\mathbf{x}_i)}{\|\boldsymbol{\beta}\|} = \frac{y_i(\alpha + \boldsymbol{\beta}^\top \mathbf{x}_i)}{\|\boldsymbol{\beta}\|}, \quad (7)$$

e queremos escolher  $\alpha$  e  $\boldsymbol{\beta}$  de modo a maximizar essa distância.

- A margem máxima é encontrada resolvendo

$$\operatorname{argmax}_{\alpha, \boldsymbol{\beta}} \left\{ \frac{1}{\|\boldsymbol{\beta}\|} \min_i [y_i(\alpha + \boldsymbol{\beta}^\top \mathbf{x}_i)] \right\}. \quad (8)$$

- A solução de (8) é complicada e sua **formulação canônica** pode ser convertida num problema mais fácil por meio do uso de **Multiplicadores de Lagrange**.

## Exemplo CMM

Consideremos os 12 pontos dispostos na Figura 1, sendo 6 em cada classe. Usando a função `svm` do pacote `e1071` e o comando `summary(svm.model)` obtemos o seguinte resultado:

Call:

```
svm(formula = type ~ ., data = my.data, type = "C-classification",
kernel = "linear", scale = FALSE)
```

Parameters:

SVM-Type: C-classification

SVM-Kernel: linear

cost: 1

gamma: 0.5

Number of Support Vectors: 3

( 1 2 )

Number of Classes: 2

Levels:

-1 1

## Exemplo CMM

- Observe que a função usa o kernel linear, que corresponde ao CMM.
- As opções *cost* e *gamma* serão explicadas adiante.
- Os coeficientes do hiperplano separador, que nesse caso é uma reta, podem ser obtidos por meio dos comandos

```
\alpha = svm.model$rho  
  
\beta = t(svm.model$coefs) %*% svm.model $ SV  
  
e são  
  
> alpha  
[1] 5.365853  
  
> beta  
  
x1          x2  
[1,] -0.8780489 -1.097561
```

## Exemplo CMM

- A equação do hiperplano separador, disposto na Figura 3 é  $5,366 - 0,878X_1 - 1,098X_2 = 0$ .
- Na mesma figura, indicamos as fronteiras de separação e os vetores suporte, dados pela solução de (4). Note que os coeficientes  $\beta_1 = 0,8780489$  e  $\beta_2 = -1,097561$  não satisfazem a restrição indicada em (4), pois foram obtidos por meio da formulação canônica do problema em que a restrição é imposta ao numerador de (7). Para detalhes, consulte a Nota de Capítulo 3.
- Neste caso há três vetores suporte (indicados por círculos azuis), um na Classe 1 (ponto em vermelho) e dois na Classe 2 (pontos em preto). Os demais pontos estão em lados separados, delimitados pelas fronteiras de separação (não há pontos entre as fronteiras).
- A margem é  $m = 0,71$ .

## Exemplo CMM

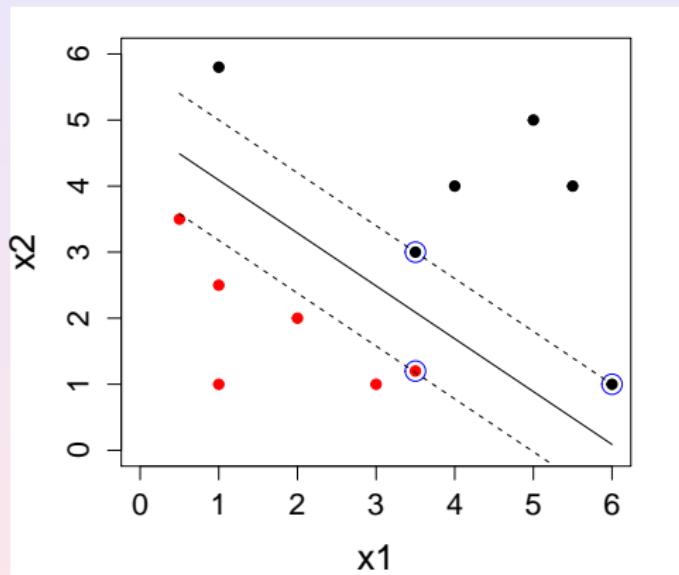


Figura 3: Hiperplano (reta) separador, margem, fronteiras e vetores suporte.

## Exemplo CMM

- Consideremos agora dois pontos,  $x_0^*$  e  $x_1^*$ , o primeiro na Classe 1 e o segundo na Classe 2 e vamos classificá-los, usando o algoritmo.
- Por meio da função `predict`, obtemos a Figura 4, que mostra a classificação correta de ambos os pontos (representados nas cores verde e azul).
- Se o problema acima não tiver solução não existirá hiperplano separador, como é o caso apresentado na Figura 2. Nesse caso precisamos recorrer a um classificador que `quase` separa as duas classes. É o que veremos a seguir.

## Exemplo CMM

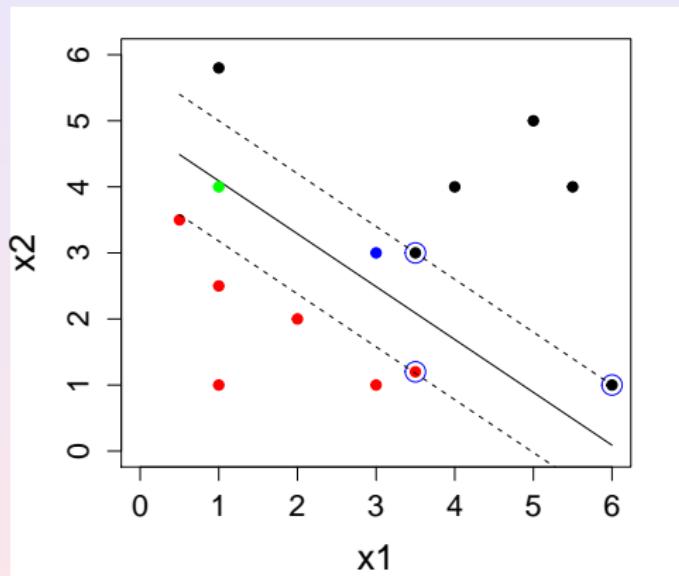


Figura 4: Classificação dos pontos indicados pelas cores verde e azul.

# CMF

- Se não existir um hiperplano separador, como aquele do Exemplo anterior, observações podem estar do lado errado da margem ou mesmo do hiperplano, correspondendo nesse caso a classificações erradas.
- O **classificador de margem flexível** (CMF), também conhecido como **classificador baseado em suporte vetorial**, é escolhido de modo a classificar corretamente a maioria das observações, o que se consegue com a introdução de **variáveis de folga**,  $\xi = (\xi_1, \dots, \xi_n)^\top$ , no seguinte problema de otimização:

$$\begin{aligned} & \text{maximizar}_{(\alpha, \beta, \xi)} \quad m(\alpha, \beta, \xi), \\ & \text{sujeito a} \end{aligned} \tag{9}$$

$$\sum_{i=1}^p \beta_i^2 = 1, \tag{10}$$

$$y_i(\alpha + \beta^\top \mathbf{x}_i) \geq m(\alpha, \beta, \xi)(1 - \xi_i), \tag{11}$$

$$\xi_i \geq 0, \quad \sum_{i=1}^n \xi_i \leq C.$$

em que  $C$  é uma constante positiva. Veja abaixo para mais detalhes sobre  $C$ .

- Embora esse tipo de classificador seja conhecido como **support vector classifier** ou **soft margin classifier**, optamos por denominá-lo “classificador de margem flexível” para diferenciá-lo do “classificador de margem máxima”, que também é baseado em vetores suporte.
- As variáveis de folga permitem que observações estejam do lado errado da margem ou do hiperplano. Pontos tais que  $\xi_i = 0$  são corretamente classificados e estão sobre a fronteira de separação ou do lado correto da fronteira. Pontos para os quais  $0 < \xi_i \leq 1$  estão dentro da fronteira da margem, mas do lado correto do hiperplano, e pontos para os quais  $\xi_i > 1$  estão do lado errado do hiperplano e serão classificados erroneamente. Veja a Figura 5, extraída de Bishop (2006). Nessa figura,  $m$  está normalizada apropriadamente, veja as Notas de Capítulo 3 e 4.
- O objetivo é maximizar a margem e, então, minimizamos

$$C \sum_{i=1}^n \xi_i + \frac{1}{2} \|\beta\|^2, \quad (12)$$

em que  $C > 0$  controla o balanço entre a penalidade das variáveis de folga e a margem.

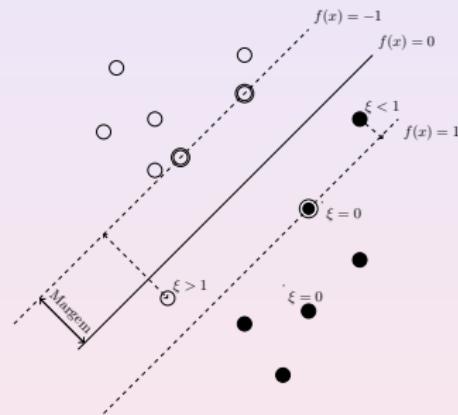


Figura 5: Detalhes sobre o classificador de margem flexível.

- Como qualquer ponto classificado erroneamente satisfaz  $\xi_i > 1$ , segue-se que  $\sum_{i=1}^n \xi_i$  é um limite superior para o número de classificações errôneas. No limite, quando  $C \rightarrow \infty$ , obtemos o CMM.
- Queremos minimizar (12) sujeito a (9). Veja a Nota de Capítulo 4.
- A constante  $C \geq 0$  deve ser escolhida apropriadamente e determina o número de violações (classificações erradas) permitidas pelo algoritmo. Se  $C = 0$ , então não há violações e  $\xi_1 = \dots = \xi_n = 0$ . Se  $C$  aumenta, a margem fica mais larga e o contrário ocorre se  $C$  decresce. O valor de  $C$  tem a ver com a relação viés–variância: quando a constante  $C$  é pequena, o viés é pequeno e a variância é grande; se  $C$  é grande, o viés é grande e a variância é pequena.

Pode-se dizer que  $C$  representa o **custo** do classificador.

- A constante  $C$  normalmente é escolhida por **validação cruzada**. O pacote [e1071](#) tem uma função, [tune\(\)](#), que realiza esse procedimento para escolher o melhor modelo, para diferentes valores de  $C$ .

## Exemplo 1 CMF

**Exemplo 1.** Consideremos agora os dados dispostos na Figura 3 em que as duas classes não são perfeitamente separáveis. Nesse caso, a utilização da função tune() do pacote e1071 gera o seguinte resultado (editado) indicando que a melhor opção é considerar  $C = 4$  e  $\gamma = 0.5$ .

```
Parameter tuning of svm:  
- sampling method: 10-fold cross validation  
- best parameters:  
  gamma   cost  
    0.5      4  
- best performance: 0.5
```

```
- Detailed performance results:  
  gamma cost error dispersion  
1   0.5     4   0.50  0.4714045  
2   1.0     4   0.60  0.4594683  
3   2.0     4   0.70  0.4216370  
4   0.5     8   0.65  0.4743416  
5   1.0     8   0.65  0.4743416  
6   2.0     8   0.70  0.4216370  
7   0.5    16   0.65  0.4743416  
8   1.0    16   0.65  0.4743416  
9   2.0    16   0.70  0.4216370
```

## Exemplo 1 CMF

Com esses parâmetros, as funções `svm` e `summary` geram o seguinte resultado, indicando que há 8 vetores suporte, 4 em cada classe.

```
svm(formula = type ~ ., data = my.data, type = "C-classification",
kernel = "linear", gamma = 0.5, cost = 4, scale = FALSE)
Parameters:
  SVM-Type: C-classification
  SVM-Kernel: linear
  cost: 4
  gamma: 0.5
Number of Support Vectors: 8
( 4 4 )
Number of Classes: 2
Levels:
-1 1
```

## Exemplo 1 CMF

Um gráfico indicando os vetores suporte e as regiões de classificação correspondentes está apresentados na Figura 6.

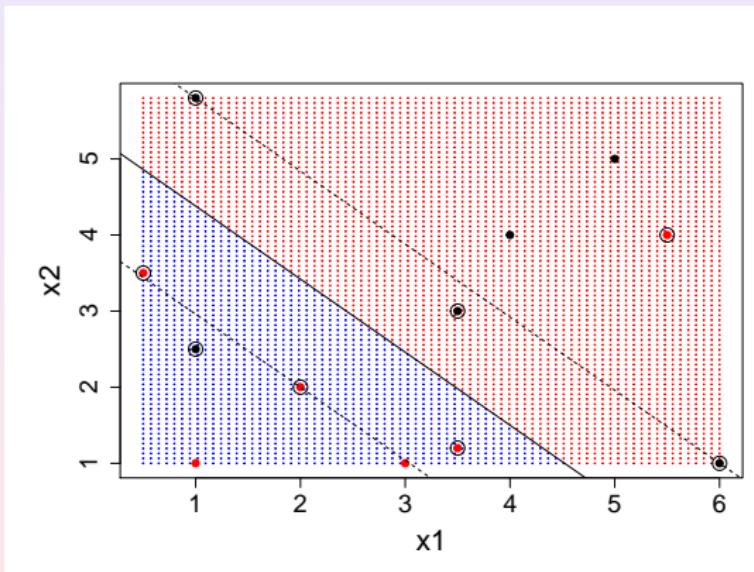


Figura 6: Vetores suporte para os dados da Figura 3.

## Exemplo 1 CMF

- A equação do hiperplano classificador é  $3,760 - 0,676x_1 - 0,704x_2 = 0$  ou equivalentemente,  $x_2 = 3,760/0,704 - 0,676/0,704x_1 = 5,339 - 0,960x_1$ . A margem correspondente é  $m = (0,676^2 + 0,704^2)^{1/2} = 0,976$ . Para detalhes, consulte as Notas de Capítulo 3 e 4.
- Com os comandos `svm.pred <- predict(svm.model, my.data)` e `table(svm.pred, ys)` podem-se obter uma tabela com as classificações certas e erradas assim como as classificações determinadas pelo algoritmo. No exemplo, há 2 classificações erradas conforme indicado na Tabela 1.

## Exemplo 1 CMF

Tabela 1: Coordenadas e classificação dos pontos do Exemplo 8.1  
com classificação predita pelo algoritmo

observação	x1	x2	y	y predito
1	0.5	3.5	1	1
2	1.0	1.0	1	1
3	1.0	2.5	-1	<b>1</b>
4	2.0	2.0	1	1
5	3.0	1.0	1	1
6	3.5	1.2	1	1
7	1.0	5.8	-1	-1
8	3.5	3.0	-1	-1
9	4.0	4.0	-1	-1
10	5.0	5.0	-1	-1
11	5.5	4.0	1	-1
12	6.0	1.0	-1	-1

## Exemplo 2 CMF

- Os dados do arquivo **tipofacial** foram extraídos de um estudo odontológico realizado pelo Dr. Flávio Cotrim Vellini. Um dos objetivos era utilizar medidas entre diferentes pontos do crânio para caracterizar indivíduos com diferentes tipos faciais, a saber, braquicéfalos, mesocéfalos e dolicocéfalos.
- O conjunto de dados contém observações de 11 variáveis em 101 pacientes. Para efeitos didáticos, utilizaremos apenas a altura facial e a profundidade facial como variáveis preditoras.
- A Figura 7 mostra os três grupos (correspondentes à classificação do tipo facial).

## Exemplo 2 CMF

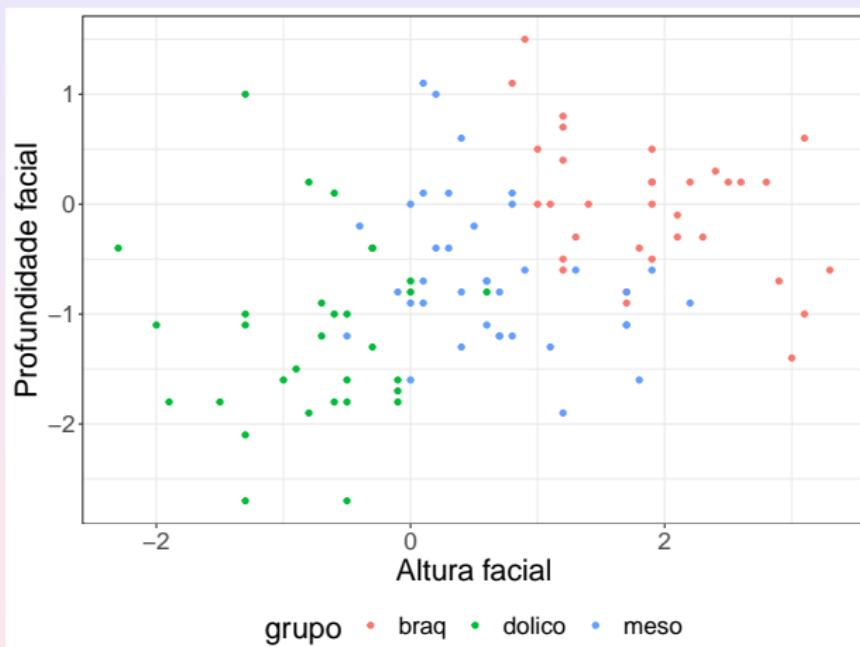


Figura 7: Gráfico de dispersão com identificação dos três tipos faciais.

## Exemplo 2 CMF

Utilizando a função `tune.svm()` do pacote `e1071` por meio dos seguintes comandos

```
> escolhaparam <- tune.svm(grupo ~ altfac + proffac, data = face,
    gamma = 2^(-2:2), cost = 2^2:5,
    na.action(na.omit(c(1, NA))))
> summary(escolhaparam)
```

obtemos os resultados, apresentados abaixo, que indicam que as melhores opções para os parâmetros  $C$  e  $\gamma$  (obtidas por meio de validação cruzada de ordem 10) para o classificador de margem flexível são  $C = 4$  e  $\gamma = 2$ .

## Exemplo 2 CMF

Parameter tuning of svm:

- sampling method: 10-fold cross validation

- best parameters:

gamma cost

2 4

- best performance: 0.1281818

- Detailed performance results:

	gamma	cost	error	dispersion
1	0.25	4	0.1481818	0.1774759
2	0.50	4	0.1681818	0.1700348
3	1.00	4	0.1681818	0.1764485
4	2.00	4	0.1281818	0.1241648
5	4.00	4	0.1581818	0.1345127
6	0.25	5	0.1481818	0.1774759
7	0.50	5	0.1681818	0.1700348
8	1.00	5	0.1481818	0.1503623
9	2.00	5	0.1281818	0.1148681
10	4.00	5	0.1772727	0.1453440

## Exemplo 2 CMF

Por intermédio da função `svm` com os parâmetros  $C = 4$  e `gamma=2` obtemos o seguinte resultado com o classificador de margem flexível:

```
svm.model <- svm(grupo ~ altfac + proffac, data = face,
                  kernel = "linear", gamma=2, cost=4)
summary(svm.model)
```

Parameters:

SVM-Type: C-classification

SVM-Kernel: linear

cost: 4

Number of Support Vectors: 43

( 12 10 21 )

Number of Classes: 3

Levels:

braq dolico meso

## Exemplo 2 CMF

A tabela de classificação obtida com os comandos apresentados abaixo, indica o número de classificações certas e erradas.

```
svm.pred <- predict(svm.model, face)
table(pred = svm.pred, true = face$grupo)
```

		true		
pred	braq	dolico	meso	
braq	26	0	2	
dolico	0	28	4	
meso	7	3	31	

Acurácia=0,84

## Exemplo 2 CMF

- Na Figura 8 apresentamos o gráfico de classificação correspondente, obtido por meio do comando

```
plot(svm.model, face, proffac ~ altfac, svSymbol = 4,  
      dataSymbol = 4, cex.lab=1.8, main="",  
      color.palette = terrain.colors)
```

- Uma das características importantes dos classificadores baseados em vetores suporte é que apenas as observações que se situam sobre a margem ou do lado errado da mesma afetam o hiperplano.
- Observações que se situam no lado correto da margem podem ser alteradas (mantendo-se suas classificações) sem que o hiperplano separador seja afetado.

## Exemplo 2 CMF

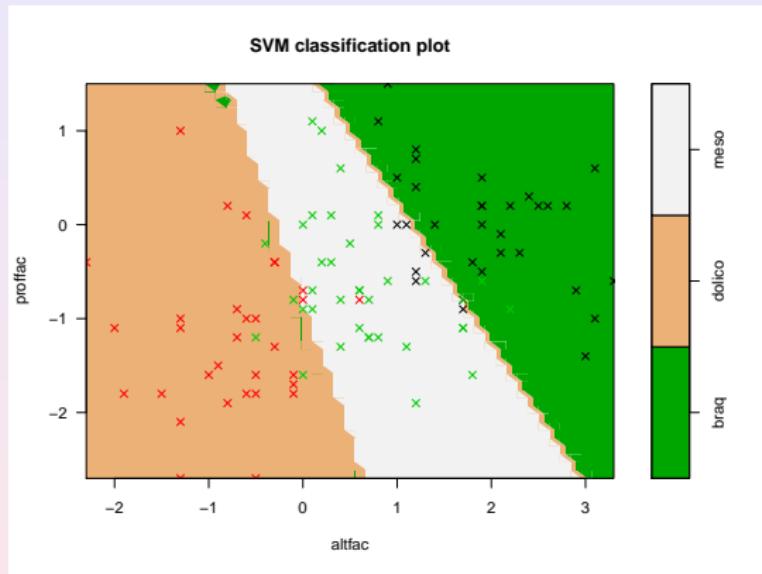


Figura 8: Classificação do tipo facial obtida pelo classificador de margem flexível.

## Referências

- Cortes, C. and Vapnik, V. (1995). Support vector networks. *Machine Learning*, **20**, 273-297.
- James, G., Witten, D., Hastie, T. and Tibshirani, R. (2017). *Introduction to Statistical Learning*. Springer.
- Morettin, P. A. e Singer, J. M. (2022). *Estatística e Ciência de Dados*. LTC.
- Vapnik, V. and Chervonenkis, A. (1964). A note on a class of perceptrons. *Automation and Remote Control*, **25**.
- Vapnik, V. and Chervonenkis, A. (1974). *Theory of Pattern recognition* [in Russian]. Moskow: Nauka.
- Vapnik, V. (1995). *The Nature of Statistical Learning Theory*. New York: Springer.
- Vapnik, V. (1998). *Statistical Learning Theory*. New York: Wiley.

# MAE 5905: Introdução à Ciência de Dados

Pedro A. Morettin

Instituto de Matemática e Estatística  
Universidade de São Paulo  
[pam@ime.usp.br](mailto:pam@ime.usp.br)  
<http://www.ime.usp.br/~pam>

## Aula 10

8 de maio de 2023

# Sumário

1 Classificador de Margem Não Linear

2 Noções da Teoria

3 Regressão por SVM

- Na seção anterior apresentamos um algoritmo de classificação (CMF), usado quando as fronteiras são lineares. Para fronteiras não lineares, precisamos aumentar a dimensão do espaço de dados por meio de outras funções, polinomiais ou não, para determinar as fronteiras de separação.
- Pode-se demonstrar que um classificador linear como aquele definido anteriormente (CMF) depende somente dos vetores suporte e pode ser escrito na forma

$$f(\mathbf{x}) = \sum_{i \in S} \gamma_i \langle \mathbf{x}, \mathbf{x}_i \rangle + \delta, \quad (1)$$

em que  $S$  indica o conjunto dos vetores suporte, os  $\gamma_i$  são funções de  $\alpha$  e  $\beta$  e  $\langle \mathbf{x}, \mathbf{y} \rangle$  indica o produto interno dos vetores  $\mathbf{x}$  e  $\mathbf{y}$ .

- Uma das vantagens de se utilizar **kernels** na construção de classificadores é que eles dependem somente dos vetores suporte e não de todas as observações o que implica uma redução considerável no custo computacional.

# CMNL

- O classificador CMF usa um *kernel* linear, da forma

$$K(\mathbf{x}_i, \mathbf{x}_j) = \sum_{k=1}^p x_{ik} x_{jk} = \mathbf{x}_i^\top \mathbf{x}_j.$$

- Se quisermos usar um CMF em um espaço característico de dimensão maior, podemos incluir polinômios de grau maior ou mesmo outras funções na definição do classificador.
- Os *kernels* mais utilizados na prática são:
  - lineares:  $K(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{x}_1^\top \mathbf{x}_2$ ;
  - polinomiais:  $K(\mathbf{x}_1, \mathbf{x}_2) = (a + \mathbf{x}_1^\top \mathbf{x}_2)^d$ ;
  - radiais:  $K(\mathbf{x}_1, \mathbf{x}_2) = \exp(-\gamma \|\mathbf{x}_1 - \mathbf{x}_2\|^2)$ , com  $\gamma > 0$  constante.
  - tangentes hiperbólicas:  $K(\mathbf{x}_1, \mathbf{x}_2) = \tanh(\theta + k \mathbf{x}_1^\top \mathbf{x}_2)$ .

Os **classificadores CMNL** são obtidos combinando-se CMF com **kernels** não lineares, de modo a obter

$$f(\mathbf{x}) = \alpha + \sum_{i \in S} \gamma_i K(\mathbf{x}, \mathbf{x}_i) + \delta. \quad (2)$$

em que os  $\gamma_i$  são funções de  $\alpha$  e  $\beta$ .

## Exemplo CMNL

- **Exemplo.** Consideremos uma análise alternativa para dados do exemplo anterior, utilizando um *kernel* polinomial, de grau 3.
- Os comandos e resultados da reanálise dos dados por meio do classificador de margem não linear são:

```
escolhaparam <- tune.svm(grupo ~ altfac + proffac, data = face,
                           kernel = "polynomial", degree=3,
                           gamma = 2^(-1:2), cost = 2^2:6)
> summary(escolhaparam)
```

Parameter tuning of svm:

- sampling method: 10-fold cross validation

- best parameters:

degree gamma cost  
3 0.5 4

- best performance: 0.1681818

## Exemplo CMNL

- Detailed performance results:

	degree	gamma	cost	error	dispersion
1	3	0.5	4	0.1681818	0.09440257
2	3	1.0	4	0.1772727	0.12024233
3	3	2.0	4	0.1872727	0.11722221
4	3	4.0	4	0.1872727	0.11722221
5	3	0.5	5	0.1972727	0.11314439
6	3	1.0	5	0.1772727	0.12024233
7	3	2.0	5	0.1872727	0.11722221
8	3	4.0	5	0.1872727	0.11722221
9	3	0.5	6	0.1872727	0.12634583
10	3	1.0	6	0.1772727	0.12024233
11	3	2.0	6	0.1872727	0.11722221
12	3	4.0	6	0.1872727	0.11722221

## Exemplo CMNL

```
svm.model <- svm(grupo ~ altfac + proffac, data=face,
                  type='C-classification', kernel='polynomial',
                  degree=3, gamma=1, cost=4, coef0=1, scale=FALSE)
summary(svm.model)
```

Parameters:

```
SVM-Type: C-classification
SVM-Kernel: polynomial
cost: 4
degree: 3
coef.0: 1
```

Number of Support Vectors: 40  
( 11 10 19 ) Number of Classes: 3

Levels:

```
braq dolico meso
```

A tabela de classificação é

	true		
pred	braq	dolico	meso
braq	29	0	4
dolico	0	26	3
meso	4	5	30

acurácia=0,84

## Exemplo CMNL

O gráfico correspondente está apresentado na Figura 1.

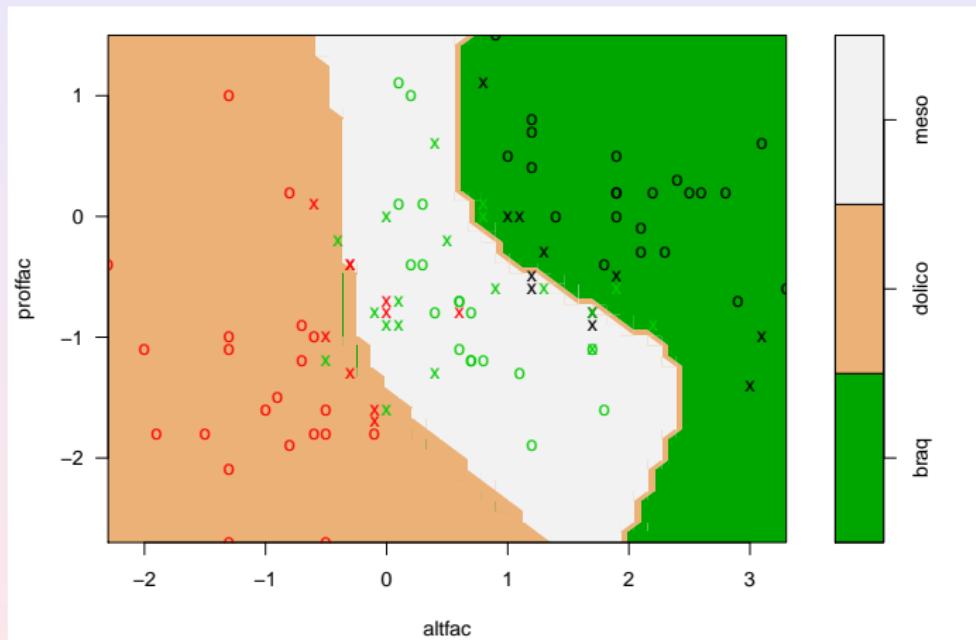


Figura: Classificação do tipo facial obtida pelo classificador de margem não linear.

## Exemplo CMNL

- Neste caso, o número de classificações erradas (16) é igual ao caso do classificador de margem flexível. A TEC é 0,16.
- Com base nesses resultados, podemos classificar indivíduos para os quais dispomos apenas dos valores das variáveis preditoras. Com essa finalidade, consideremos o seguinte conjunto de previsão com 4 indivíduos:

	paciente	altfac	proffac
1	102	1.4	1.0
2	103	3.2	0.1
3	104	-2.9	-1.0
4	105	0.5	0.9

# CMNL

Por meio dos seguintes comandos

```
svm.model <- svm(grupo ~ altfac + proffac, data=face, type='C-classification',
                  kernel='polynomial', degree=3, gamma=1, cost=4, coef0=1,
                  scale=FALSE, probability=TRUE)
prednovos <- predict(svm.model, teste, probability=TRUE)
```

obtemos a tabela com as probabilidades de classificação de cada um dos 4 indivíduos

```
 1      2      3      4
braq    braq   dolico  meso
attr(),"probabilities

braq      dolico      meso
1 0.954231749 0.0193863931 0.0263818582
2 0.961362058 0.0006154201 0.0380225221
3 0.008257919 0.9910764215 0.0006656599
4 0.254247666 0.1197179567 0.6260343773
```

Levels: braq dolico meso

O processo classifica os indivíduos 102 e 103 como braquicéfalos, o indivíduo 104 como dolicocéfalo e o 105, como mesocéfalo.

## Hiperplano separador

Um hiperplano definido num espaço de dimensão  $p$  é um **subespaço** de dimensão  $p - 1$  definido por

$$\alpha + \beta_1 X_1 + \dots + \beta_p X_p = 0. \quad (3)$$

Um ponto com coordenadas  $(x_1, \dots, x_p)$  satisfazendo (3) situa-se no hiperplano. Se  $\alpha + \beta_1 x_1 + \dots + \beta_p x_p > 0$ , esse ponto situa-se num lado do hiperplano e se  $\alpha + \beta_1 x_1 + \dots + \beta_p x_p < 0$ , o ponto situa-se no outro lado desse hiperplano. Dessa forma, o hiperplano separa o espaço  $p$  dimensional em duas metades.

## Teoria-CMM

- Consideremos o espaço característico  $\mathcal{T} = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$  e as respostas  $y_1, \dots, y_n$  com  $y_i \in \{-1, 1\}$ , definindo o conjunto de treinamento. Novos dados  $\mathbf{x}_0$  são classificados de acordo com o sinal de  $f(\mathbf{x}_0)$ .
- Suponha que exista um hiperplano separador, de modo que  $\alpha$  e  $\beta$  são tais que  $f(\mathbf{x}) > 0$ , para pontos com  $y = +1$  e  $f(\mathbf{x}) < 0$ , para pontos com  $y = -1$ , de modo que  $yf(\mathbf{x}) > 0$ , para qualquer dado de treinamento.
- O CMM tem como objetivo maximizar a margem que é a menor distância entre o hiperplano e qualquer ponto do conjunto de treinamento.
- Para entender o procedimento de otimização, considere a distância de um ponto  $\mathbf{x}$  ao hiperplano cuja equação é  $f(\mathbf{x}) = 0$ , nomeadamente

$$d = |f(\mathbf{x})|/\|\beta\|,$$

em que denominador indica a norma do vetor  $\beta$ .

## Teoria-CMM

- Como o interesse está nos pontos que são corretamente classificados, devemos ter  $y_i f(\mathbf{x}_i) > 0$ ,  $i = 1, \dots, n$ . Logo, a distância entre qualquer ponto  $\mathbf{x}_i$  e o hiperplano é

$$\frac{y_i f(\mathbf{x}_i)}{\|\beta\|} = \frac{y_i(\alpha + \beta^\top \mathbf{x}_i)}{\|\beta\|}. \quad (4)$$

- A margem é a distância do hiperplano ao ponto  $\mathbf{x}$  mais próximo e queremos escolher  $\alpha$  e  $\beta$  de modo a maximizar essa distância. A margem máxima é obtida por meio da resolução de

$$\operatorname{argmax}_{\alpha, \beta} \left\{ \frac{1}{\|\beta\|} \min \left[ y_i(\alpha + \beta^\top \mathbf{x}_i) \right] \right\}. \quad (5)$$

- A solução de (5) é complicada mas é possível obtê-la por meio da utilização de **Multiplicadores de Lagrange**. Note que se multiplicarmos  $\alpha$  e  $\beta$  por uma constante, a distância de um ponto  $\mathbf{x}$  ao hiperplano separador não se altera.

## Teoria-CMM

- Logo podemos considerar a transformação  $\alpha^* = \alpha/f(\mathbf{x})$  e  $\beta^* = \beta/f(\mathbf{x})$  e para o ponto mais próximo do hiperplano, digamos  $\mathbf{x}^*$ , obtendo

$$y^*(\alpha + \beta^\top \mathbf{x}^*) = 1, \quad (6)$$

e consequentemente,  $d = ||\beta||^{-1}$ .

- Desse modo, todos os pontos do conjunto de treinamento satisfarão

$$y_i(\alpha + \beta^\top \mathbf{x}_i) \geq 1, \quad i = 1, \dots, n. \quad (7)$$

Esta relação é chamada **representação canônica do hiperplano separador**.

- Dizemos que há uma **restrição ativa** para os pontos em que há igualdade; para os pontos em que vale a desigualdade, dizemos que há uma **restrição inativa**. Como sempre haverá um ponto que está mais próximo do hiperplano, sempre haverá uma restrição ativa.

## Teoria-CMM

- Então, o problema de otimização implica maximizar  $\|\beta\|^{-1}$ , que é equivalente a minimizar  $\|\beta\|^2$ .
- Na linguagem de Vapnik (1995), isso equivale a escolher  $f(\mathbf{x})$  de maneira que seja a mais achatada (*flat*) possível, que por sua vez implica que  $\beta$  deve ser pequeno.
- Isso corresponde à resolução do problema de **programação quadrática**

$$\operatorname{argmin}_{\alpha, \beta} \left\{ \frac{1}{2} \|\beta\|^2 \right\}, \quad (8)$$

sujeito a (7). O fator  $1/2$  é introduzido por conveniência.

- Com esse objetivo, para cada restrição em (7), introduzimos os Multiplicadores de Lagrange  $\lambda_i \geq 0$ , obtendo a função lagrangeana

$$L(\alpha, \beta, \lambda) = \frac{1}{2} - \sum_{i=1}^n \lambda_i [y_i(\alpha + \beta^\top \mathbf{x}_i) - 1], \quad (9)$$

em que  $\lambda = (\lambda_1, \dots, \lambda_n)^\top$ . O sinal negativo no segundo termo de (9) justifica-se por que queremos minimizar em relação a  $\alpha$  e  $\beta$  e maximizar em relação a  $\lambda$ .

## Teoria-CMM

- Derivando  $L$  em relação a  $\beta$  e a  $\lambda$ , obtemos

$$\beta = \sum_{i=1}^n \lambda_i y_i \mathbf{x}_i \quad \text{e} \quad \sum_{i=1}^n \lambda_i y_i = 0. \quad (10)$$

- Eliminando  $\alpha$  e  $\beta$  em (9) e usando (10), obtemos a chamada **representação dual** do problema da margem máxima, no qual maximizamos

$$\tilde{L}(\lambda) = \sum_{i=1}^n \lambda_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \lambda_i \lambda_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j), \quad (11)$$

com respeito a  $\lambda$ , sujeito às restrições

$$\lambda_i \geq 0, \quad i = 1, \dots, n, \quad (12)$$

$$\sum_{i=1}^b \lambda_i y_i = 0. \quad (13)$$

- Em (11),  $K(\mathbf{x}, \mathbf{y}) = \mathbf{x}^\top \mathbf{y}$  é um *kernel* linear, que será estendido para algum *kernel* mais geral com a finalidade de ser aplicado a espaços característicos cuja dimensionalidade excede o número de dados. Esse *kernel* deve ser positivo definido.

## Teoria-CMM

- Para classificar um novo dado  $\mathbf{x}_0$  usando o modelo treinado, avaliamos o sinal de  $f(\mathbf{x}_0)$ , que por meio de (10), pode ser escrito como

$$f(\mathbf{x}_0) = \alpha + \sum_{i=1}^n \lambda_i y_i K(\mathbf{x}_0, \mathbf{x}_i). \quad (14)$$

- Pode-se demonstrar (veja Bishop, 2006), que esse tipo de otimização restrita satisfaz certas condições, chamadas de **condições de Karush-Kuhn-Tucker** (KKT) que implicam

$$\begin{aligned} \lambda_i &\geq 0, \\ y_i f(\mathbf{x}_i) - 1 &\geq 0, \\ \lambda_i(y_i f(\mathbf{x}_i) - 1) &= 0. \end{aligned} \quad (15)$$

## Teoria-CMM

- Para cada ponto, ou  $\lambda_i = 0$  ou  $y_i f(\mathbf{x}_i) = 1$ . Um ponto para o qual  $\lambda_i = 0$  não aparece em (14) não tem influência na classificação de novos pontos.
- Os pontos restantes são chamados **vetores suporte** e satisfazem  $y_i f(\mathbf{x}_i) = 1$ ; logo esses pontos estão sobre as fronteiras do espaço separador, como na Figura 3 da Aula 9.
- O valor de  $\alpha$  pode ser encontrado a partir de

$$y_i \left( \sum_{j \in S} \lambda_j y_j K(\mathbf{x}_j, \mathbf{x}_i) + \alpha \right) = 1, \quad (16)$$

em que  $S$  é o conjunto dos vetores suporte.

- Multiplicando essa expressão por  $y_i$ , observando que  $y_i^2 = 1$  e tomado a média de todas as equações sobre  $S$ , obtemos

$$\alpha = \frac{1}{n_S} \sum_{i \in S} \left( y_i - \sum_{j \in S} \lambda_j y_j K(\mathbf{x}_i, \mathbf{x}_j) \right), \quad (17)$$

em que  $n_S$  é o número de vetores suporte.

## Teoria–CMF

- Vamos considerar agora, o caso em que as duas classes podem se sobrepor. Precisamos modificar o CMM para permitir que alguns pontos do conjunto de treinamento sejam classificados erroneamente. Para isso introduzimos uma penalidade, que cresce com a distância ao hiperplano separador.
- Isso é conseguido pela introdução de **variáveis de folga** (*slack*)  $\xi_i \geq 0, i = 1, \dots, n$ , uma para cada dado.
- Então,  $\xi_i = 0$  para pontos sobre ou dentro da fronteira correta [delimitada por  $f(\mathbf{x}) = -1$  e  $f(\mathbf{x}) = 1$ ] e  $\xi_i$  dado pela distância do ponto à fronteira, para os outros pontos.
- Assim, um ponto que estiver sobre o hiperplano  $f(\mathbf{x}) = 0$  terá  $\xi_i = 1$  e pontos com  $\xi_i > 1$  são classificados erroneamente.
- Nesse caso, a restrição para o caso CMM será substituída por

$$y_i(\alpha + \beta^\top \mathbf{x}_i) \geq 1 - \xi_i, \quad i = 1, \dots, n, \quad (18)$$

com  $\xi_i \geq 0$ .

## Teoria - CMF

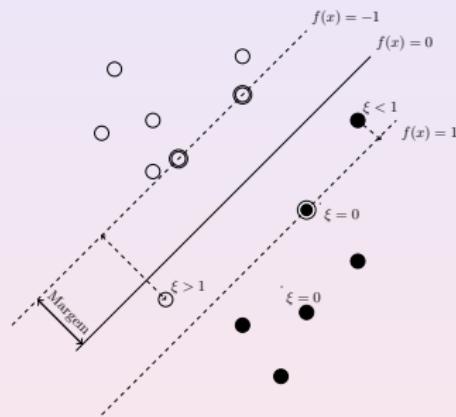


Figura: Detalhes sobre o classificador de margem flexível.

## Teoria-CMF

- Pontos para os quais  $0 < \xi_i \leq 1$  estão dentro da fronteira da margem, mas do lado correto do hiperplano, e pontos para os quais  $\xi_i > 1$  estão do lado errado do hiperplano e são classificados erroneamente. Pontos para os quais  $\xi_i = 0$  são corretamente classificados e estão sobre a fronteira da margem ou do lado correto da fronteira da margem.
- Nesse contexto, estamos diante de uma **margem flexível** ou **suave**. O objetivo é maximizar a margem e, para isso, minimizamos

$$C \sum_{i=1}^n \xi_i + \frac{1}{2} \|\beta\|^2, \quad (19)$$

em que  $C > 0$  controla o balanço entre a penalidade das variáveis de folga e a margem.

- Como qualquer ponto classificado erroneamente satisfaz  $\xi_i > 1$ , segue-se que  $\sum_{i=1}^n \xi_i$  é um limite superior do número de classificações errôneas. No limite, quando  $C \rightarrow \infty$ , obtemos o CMM.

## Teoria–CMF

- Para minimizar (19) sujeito a (18) e  $\xi_i > 0$  consideramos o lagrangeano

$$L(\alpha, \beta, \mathbf{x}_i, \lambda, \mu) = \frac{1}{2} \|\beta\|^2 + C \sum_{i=1}^n \xi_i - \sum_{i=1}^n \lambda_i [y_i f(\mathbf{x}_i) + \xi_i - 1] - \sum_{i=1}^n \mu_i \xi_i, \quad (20)$$

em que  $\lambda_i \geq 0, \mu_i \geq 0$  são multiplicadores de Lagrange.

- Derivando (21) com relação a  $\beta, \alpha, \xi_i$ , obtemos

$$\beta = \sum_{i=1}^n \lambda_i y_i \mathbf{x}_i, \quad \sum_{i=1}^n \lambda_i y_i = 0 \quad (21)$$

e

$$\lambda_i = C - \mu_i. \quad (22)$$

## Teoria - CMF

- Substituindo (21) - (22) em (21), temos

$$\tilde{L}(\lambda) = \sum_{i=1}^n \lambda_i - \frac{1}{2} \sum_i \sum_j \lambda_i \lambda_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j), \quad (23)$$

que é uma expressão idêntica ao caso separável, com exceção das restrições, que são diferentes.

- Como  $\lambda_i \geq 0$  são multiplicadores de Lagrange e como  $\mu_i \geq 0$ , de (22) segue que  $\lambda_i \leq C$ . Logo, precisamos maximizar (23) com respeito às variáveis duais  $\lambda_i$ , sujeito a

$$0 \leq \lambda_i \leq C, \quad (24)$$

$$\sum_{i=1}^n \lambda_i y_i = 0, \quad i = 1, \dots, n. \quad (25)$$

- Novamente, estamos diante de um problema de programação quadrática.

## Teoria - CMF

- A previsão para um novo ponto  $\mathbf{x}$  é obtida avaliando o sinal de  $f(\mathbf{x})$  na equação do hiperplano(eq. 3, Aula 9). Substituindo (21) nessa mesma equação, obtemos

$$f(\mathbf{x}) = \alpha + \sum_{i=1}^n \lambda_i y_i K(\mathbf{x}, \mathbf{x}_i). \quad (26)$$

- Dados para os quais  $\lambda_i = 0$  não contribuem para (26). Os dados restantes formam os vetores de suporte. Para esses,  $\lambda_i > 0$  e, por (28) abaixo, devem satisfazer

$$y_i f(\mathbf{x}_i) = 1 - \xi_i. \quad (27)$$

- No caso de CMF, as condições de KKT são dadas por

$$\lambda_i \geq 0, \quad y_i f(\mathbf{x}_i) - 1 + \xi_i \geq 0,$$

$$\lambda_i(y_i f(\mathbf{x}_i) - 1 + \xi_i) = 0, \quad (28)$$

$$\mu_i \geq 0, \quad \xi_i \geq 0,$$

$$\mu_i \xi_i = 0, \quad i = 1, \dots, n. \quad (29)$$

## Teoria - CMF

- Procedendo como no caso de CMM, obtemos

$$\alpha = \frac{1}{N_{\mathcal{M}}} \sum_{i \in \mathcal{M}} \left( y_i - \sum_{j \in S} \lambda_j y_j K(\mathbf{x}_i, \mathbf{x}_j) \right), \quad (30)$$

em que  $\mathcal{M}$  é o conjunto do pontos tais que  $0 < \lambda_i < C$ .

- Se  $\lambda_i < C$ , então, por (22),  $\mu_i > 0$  e por (29), temos  $\xi = 0$  e tais pontos estão na fronteira de separação. Pontos com  $\lambda_i = C$  estão dentro da fronteira de separação e podem ser classificados corretamente se  $\xi_i \leq 1$  e erroneamente se  $\xi_i > 1$ .

## Teoria-CMNL

- Seja  $\mathcal{X}$  o conjunto de dados (ou de **padrões**). A função  $K : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$  é um **kernel** se existir um espaço vetorial com produto interno,  $\mathcal{H}$  (usualmente um espaço de Hilbert) e uma aplicação  $\Phi : \mathcal{X} \rightarrow \mathcal{H}$ , tal que, para todos  $x, y \in \mathcal{X}$ , tivermos

$$K(x, y) = \langle \Phi(x), \Phi(y) \rangle. \quad (31)$$

$\Phi$  é a **aplicação característica** e  $\mathcal{H}$ , o **espaço característico**.

- Por exemplo, tomemos  $\mathcal{X} = \mathbb{R}^2$  e  $\mathcal{H} = \mathbb{R}^3$  e definamos

$$\begin{aligned}\Phi : \mathbb{R}^2 &\rightarrow \mathbb{R}^3, \\ (x_1, x_2) &\rightarrow (x_1^2, x_2^2, \sqrt{2}x_1x_2).\end{aligned}$$

Então, se  $x = (x_1, x_2)$  e  $y = (y_1, y_2)$ , é fácil verificar que  $\langle \Phi(x), \Phi(y) \rangle = \langle x, y \rangle$ ; logo  $K(x, y) = \langle \Phi(x), \Phi(y) \rangle = \langle x, y \rangle$  é um **kernel**.

## Teoria-CMNL

- Para tornar o algoritmo de suporte vetorial não linear, notamos que ele depende somente de produtos internos entre os vetores de  $\mathcal{X}$ ; logo, é suficiente conhecer  $K(\mathbf{x}, \mathbf{x}^\top) = \langle \Phi(\mathbf{x}), \Phi(\mathbf{x}^\top) \rangle$ , e não  $\Phi$  explicitamente. Isso permite formular o problema de otimização, substituindo a derivada do Lagrangeano no caso de CMF por

$$\beta = \sum_{i=1}^n \alpha_i \Phi(\mathbf{x}_i). \quad (32)$$

- Agora,  $\beta$  não é mais dado explicitamente como antes. Também, o problema de otimização é agora realizado no espaço característico e não em  $\mathcal{X}$ .
- Os *kernels* a serem usados têm que satisfazer certas condições de admissibilidade. Veja Smola e Schölkopf (2004) para detalhes. Os *kernels* mencionados anteriormente são admissíveis.

## Regressão via SVM

- Dado um conjunto de treinamento,  $\mathcal{T} = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)\}$ , o objetivo é obter uma função  $f(\mathbf{x}_i)$ , a mais achatada (*flat*) possível tal que  $|y_i - f(\mathbf{x}_i)| < \epsilon$ ,  $i = 1, \dots, n$  em que  $\epsilon > 0$  é o maior erro que estamos dispostos a cometer. Por exemplo,  $\epsilon$  pode ser a máxima perda que admitimos ao negociar com ações dadas certas características obtidas do balanço de um conjunto de empresas.
- No caso de funções lineares, o objetivo é determinar  $\alpha$  e  $\beta$  tais que  $|f(\mathbf{x}_i)| = |\alpha + \beta^\top \mathbf{x}_i| \leq \epsilon$ . A condição de que  $f(\mathbf{x})$  seja a mais achatada possível corresponde a que  $\beta$  seja pequeno, ou seja o problema a resolver pode ser expresso como

$$\text{minimizar } \frac{1}{2} \|\beta\|^2 \text{ sujeito a } \begin{cases} y_i - \beta^\top \mathbf{x}_i - \alpha \leq \epsilon, \\ \alpha + \beta^\top \mathbf{x}_i - y_i \leq \epsilon \end{cases}. \quad (33)$$

## Regressão via SVM

- Nem sempre as condições (33) podem ser satisfeitas e nesse caso, assim como nos modelos de classificação, podemos introduzir variáveis de folga  $\xi_i$  e  $\xi_i^*$ ,  $i = 1, \dots, n$ , que permitem flexibilizar a restrição de que o máximo erro permitido seja  $\epsilon$ . O problema a resolver nesse contexto é

$$\text{minimizar } \frac{1}{2} \|\beta\|^2 + \sum_{i=1}^n C(\xi + \xi^*) \text{ sujeito a } \begin{cases} y_i - \beta^\top \mathbf{x}_i - \alpha \leq \epsilon + \xi_i, \\ \alpha + \beta^\top \mathbf{x}_i - y_i \leq \epsilon + \xi_i^*, \\ \xi_i, \xi_i^* > 0. \end{cases} \quad (34)$$

- A constante  $C > 0$  determina um compromisso entre o achatamento da função  $f$  e o quanto estamos dispostos a tolerar erros com magnitude maior do que  $\epsilon$ .

## Regressão via SVM

- As soluções de (33) ou (34) podem ser encontradas mais facilmente usando a formulação dual (ver Nota de Capítulo 3). No caso de modelos lineares, a previsão para um elemento com valor das variáveis preditoras igual a  $\mathbf{x}_0$  é obtida de

$$f(\mathbf{x}_0) = \sum_{i=1}^n \hat{\lambda}_i K(\mathbf{x}_0, \mathbf{x}_i) + \hat{\alpha},$$

em que  $\hat{\lambda}_i$  são multiplicadores de Lagrange,  $K(\mathbf{x}_0, \mathbf{x}_i)$  é um *kernel*,  $\hat{\alpha} = y_i - \varepsilon - \hat{\beta}^\top \mathbf{x}_i$  e  $\hat{\beta} = \sum_{i=1}^n \hat{\lambda}_i \mathbf{x}_i$ .

- Os vetores suporte são aqueles para os quais os multiplicadores de Lagrange  $\hat{\lambda}_i$  são positivos.
- Se optarmos por um *kernel* linear,  $K(\mathbf{x}, \mathbf{x}_i) = \langle \mathbf{x}, \mathbf{x}_i \rangle$ .

## Regressão via SVM - Exemplo

Consideremos os dados de **distancia** com o objetivo de estudar a relação entre a distância com que motoristas conseguem distinguir um certo objeto e sua idade. O diagrama de dispersão e a reta de mínimos quadrados ajustada ( $y = 174,2 - 1,0x$ ) correspondentes estão apresentados na Figura 3.

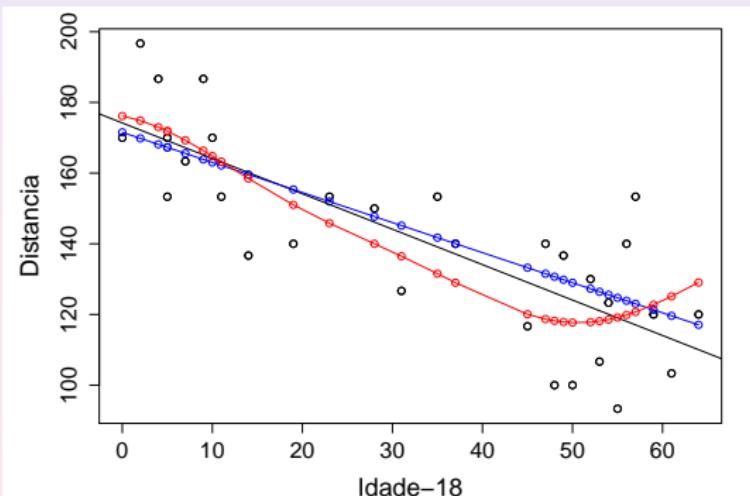


Figura 3: Regressão SVM para os dados de distância.

## Regressão via SVM - Exemplo

O ajuste de uma regressão com suporte vetorial baseada num *kernel* linear com os parâmetros *default* pode ser obtido por meio dos comandos

```
model1<- svm(x, y, kernel="linear")
summary(model1)
```

Parameters:

SVM-Type: eps-regression

SVM-Kernel: linear

cost: 1

gamma: 1

epsilon: 0.1

Number of Support Vectors: 23

```
betahat <- model1$rho
```

```
[1] -0.08572489
```

```
coef1 <- sum(model1$coefs*x[model1$index])
```

```
alfahat <- coef1/model1$rho
```

```
[1] 172.8264
```

de forma que a função previsora corresponde à  $f(x) = 172,9 - 0,09x$ .

## Regressão via SVM - Exemplo

- A previsão para as distâncias segundo esse modelo pode ser obtida por meio do comando `yhat1 <- predict(model1, x)`. O *RMSE* correspondente pode ser obtido por meio do comando `rmse(yhat1, y)` é 16,51464 (maior do que o *RMSE* associado ao ajuste por meio de mínimos quadrados, que é 16,02487).
- Um modelo mais flexível pode ser ajustado com um *kernel* radial do tipo  $K(\mathbf{x}_1, \mathbf{x}_2) = \exp(-\gamma \|\mathbf{x}_1 - \mathbf{x}_2\|^2)$  com  $\gamma > 0$  constante. Nesse caso, convém realizar uma análise de sensibilidade com validação cruzada para a seleção da melhor combinação dos valores do máximo erro  $\epsilon$  que estamos dispostos a cometer e do custo de penalização,  $C$ . Isso pode ser concretizado por meio dos comandos

```
sensib <- tune(svm, y ~ x, ranges = list(epsilon = seq(0,1,0.1),  
                                             cost = 2^(2:9)))
```

Parameter tuning of svm:

- sampling method: 10-fold cross validation

- best parameters:

epsilon cost

0.8 8

- best performance: 275.8086

## Regressão via SVM - Exemplo

Com esses resultados, realizamos um ajuste por meio de um *kernel* radial com parâmetros  $C = 8$  e  $\epsilon = 0.8$ , obtendo

```
model2 <- svm(x, y, kernel="radial", cost=8, epsilon=0.8)
summary(model2)
```

Parameters:

SVM-Type: eps-regression

SVM-Kernel: radial

cost: 8

gamma: 1

epsilon: 0.8

Number of Support Vectors: 6

O RMSE para esse modelo é 15,84272, menor do que aqueles obtidos por meio dos demais ajustes. Um gráfico com os ajustes por mínimos quadrados (em preto) e por regressões com suporte vetorial baseadas em *kernels* linear (em azul) e radial (em vermelho) está apresentado na Figura 3.

## Regressão via SVM - Observações

- Algoritmos de suporte vetorial no contexto de regressão também podem ser utilizados com o mesmo propósito de suavização daquele concretizado pelo método **Lowess** (veja a Nota de Capítulo 2 do Capítulo 5).
- Nesse contexto, a suavidade do ajuste deve ser modulada pela escolha do parâmetro  $\epsilon$ . Valores de  $\epsilon$  pequenos (próximos de zero) geram curvas mais suaves e requerem muitos vetores suporte, podendo produzir sobreajuste. Valores de  $\epsilon$  grandes (próximos de 1,0, por exemplo) geram curvas menos suaves e requerem menos vetores suporte.
- O parâmetro  $C$  tem influência no equilíbrio entre as magnitudes da margem e das variáveis de folga. Em geral, o valor desse parâmetro deve ser selecionado por meio de uma análise de sensibilidade concretizada por validação cruzada.

## Referências

- Morettin, P. A. e Singer, J. M. (2022). *Estatística e Ciência de Dados*. LTC: Rio de Janeiro.
- Smola, A. J. and Schölkopf, B. (2004). A tutorial on support vector regression. *Statistics and Computing*, **14**, 199–222.
- Vapnik, V. (1995). *The Nature of Statistical Learning Theory*. New York: Springer.
- Vapnik, V. (1998). *Statistical Learning Theory*. New York: Wiley.

# MAE 5905: Introdução à Ciência de Dados

Pedro A. Morettin

Instituto de Matemática e Estatística  
Universidade de São Paulo  
[pam@ime.usp.br](mailto:pam@ime.usp.br)  
<http://www.ime.usp.br/~pam>

## Aula 11

12 de maio de 2023

# Sumário

1 Classificação por meio de árvores

2 Bagging

3 Boosting

4 Florestas Aleatórias

5 Árvores para regressão

## Preliminares I

- Modelos baseados em árvores foram desenvolvidos por Leo Breiman e associados e são bastante utilizados tanto para classificação quanto para previsão.
- Esses modelos são baseados numa segmentação do espaço gerado pelas variáveis explicativas (preditoras) em algumas regiões em que ou a moda (no caso de variáveis respostas categorizadas) ou a média (no caso de variáveis contínuas) é utilizada como predição.
- A definição dessas regiões é baseada em alguma medida de erro de previsão (ou de classificação). Em geral, as árvores são construídas a partir de um conjunto de **observações de treinamento** e testadas em um conjunto de **observações de teste**
- Os modelos de árvores de decisão são conceitualmente e computacionalmente simples e bastante populares em função de sua interpretabilidade, apesar de serem menos precisos que outros modelos mais complexos.

## Preliminares II

- Generalizações dos modelos originais, conhecidos como **florestas aleatórias** (*random forests*) costumam apresentar grande precisão, mesmo quando comparados com modelos lineares, porém pecam pela dificuldade de interpretação. A referência básica para esse tópico é Breiman et al. (1984). O texto de Hastie and Tibshirani (1990) também contém resultados sobre esse tópico.
- Para predizer o valor de uma variável resposta  $Y$  (no caso de variáveis contínuas) ou classificar as observações em uma de suas categorias (no caso de variáveis categorizadas) a partir de um conjunto de variáveis preditoras  $X_1, \dots, X_p$ , o algoritmo usado na construção de árvores de decisão consiste essencialmente na determinação das regiões (retângulos mutuamente exclusivos) em que o espaço das variáveis preditoras é particionado. A metodologia desenvolvida em Breiman et al. (1994), e designada **CART** (de *Classification And Regression Trees*) é baseada na seguinte estratégia:
  - (a) Considere uma partição do espaço das variáveis preditoras (conjuntos dos possíveis valores de  $X_1, \dots, X_p$ ) em  $M$  regiões,  $R_1, \dots, R_M$ .

## Preliminares III

- (b) Para cada observação pertencente a  $R_j$ , o previsor (ou categoria) de  $Y$  (que designaremos  $\widehat{Y}_{R_j}$ ) será a moda (no caso discreto), a média (no caso contínuo) ou a porcentagem (no caso categorizado) dos valores de  $Y$  correspondentes àqueles com valores de  $X_1, \dots, X_p$  em  $R_j$ .
- Embora a partição do espaço das variáveis preditoras seja arbitrária, usualmente ela é composta por retângulos  $p$ -dimensionais que devem ser construídos de modo a minimizar alguma medida de erro de previsão ou de classificação (que explicitaremos posteriormente).
  - Como esse procedimento geralmente não é computacionalmente factível dado o número de partições possíveis, mesmo com  $p$  moderado, usa-se uma **divisão binária recursiva** (*recursive binary splitting, RBS*) que é uma abordagem **top-down and greedy**, segundo James et al. (2013).
  - Dado o vetor de variáveis preditoras  $\mathbf{X} = (X_1, \dots, X_p)^\top$ , o algoritmo consiste dos passos:
    - (i) Selecione uma variável preditora  $X_j$  e um limiar (ou ponto de corte)  $t$ , de modo que a divisão do espaço das variáveis preditoras nas regiões  $\{\mathbf{X} : X_j < t\}$  e  $\{\mathbf{X} : X_j \geq t\}$  corresponda ao menor erro de previsão (ou de classificação).

## Preliminares IV

- (ii) Para todos os pares  $(j, t)$ , considere as regiões

$$R_1(j, t) = \{\mathbf{X} : X_j < t\}, \quad R_2(j, t) = \{\mathbf{X} : X_j \geq t\}$$

e encontre o par  $(j, s)$  que minimize o erro de predição (ou de classificação) adotado.

- (iii) Repita o procedimento, agora dividindo uma das duas regiões encontradas, obtendo três regiões; depois divida cada uma dessas três regiões minimizando o erro de predição (ou de classificação);
  - (iv) Continue o processo até que algum critério de parada seja satisfeito.
- Um possível critério de parada consiste na obtenção de um número mínimo fixado de observações em cada região.

## Árvores para Classificação

- Quando a variável resposta  $Y$  é categorizada, o objetivo é identificar a classe mais provável (**classe modal**) associada aos valores  $\mathbf{X} = (X_1, \dots, X_p)^\top$  das variáveis preditoras.
- Neste caso, uma medida de erro de classificação, comumente denominada **taxa de erros de classificação** ( $TEC$ ) é a proporção de observações do conjunto de treinamento que não pertencem à classe modal.
- Outras medidas de erro de classificação, como entropia ou índice de Gini também podem ser usadas.
- Admitamos que a variável resposta tenha  $K$  classes e que o espaço de variáveis preditoras seja particionado em  $M$  regiões. Designando por  $\hat{p}_{mk}$ , a proporção de observações de treinamento da  $m$ -ésima região,  $m = 1, \dots, M$ , pertencentes à  $k$ -ésima classe  $k = 1, \dots, K$ , a taxa de erros de classificação dos elementos pertencentes à  $m$ -ésima região é

$$TEC_m = 1 - \max_k(\hat{p}_{mk}).$$

## Árvores para Classificação

- Como alternativa para as taxas de erros de classificação, pode-se usar o **índice de Gini** definido como

$$G_m = \sum_{k=1}^K \hat{p}_{mk} (1 - \hat{p}_{mk}),$$

que, essencialmente, corresponde à soma das variâncias das proporções de classificação em cada classe. Quando o valor de  $\hat{p}_{mk}$  para um dado nó  $m$  estiver próximo de 1 para uma dada categoria  $k$  e estiver próximo de zero para as demais categorias, o índice de Gini correspondente estará próximo de zero, indicando que para esse nó, uma das  $K$  categorias concentrará uma grande proporção dos elementos do conjunto de dados; poucos deles serão classificadas nas demais  $K - 1$  categorias. Quanto mais concentrados em uma categoria forem as classificações em um dado nó, tanto maior será o seu grau de **pureza**.

- Outra medida utilizada com o mesmo propósito e que tem características similares àquelas do coeficiente de Gini é a **entropia cruzada**, definida como

$$ET_m = \sum_{k=1}^K \hat{p}_{mk} \log(\hat{p}_{mk}).$$

## Árvores–Exemplo 1

- Vários pacotes, `tree`, `partykit`, `rpart`, `caret` podem ser utilizados com esse propósito. Cada um desses pacotes é regido por parâmetros que controlam diferentes aspectos da construção das árvores. Consultar os manuais correspondentes para nortear uma seleção adequada a problemas específicos.
- **Exemplo 1.** Consideremos novamente os dados analisados no Exemplo 9.2, disponíveis no arquivo `tipofacial`, extraídos de um estudo cujo objetivo era avaliar se duas ou mais medidas ortodônticas poderiam ser utilizadas para classificar indivíduos segundo o tipo facial (braquicéfalo, mesocéfalo ou dolicocéfalo).  
Como no Exemplo 9.2, para efeitos didáticos consideramos apenas duas variáveis preditoras, correspondentes a duas distâncias de importância ortodôntica, nomeadamente, a altura facial (`altfac`) e a profundidade facial (`proffac`).

## Árvores–Exemplo 1

- Os comandos do pacote `partykit` (com os parâmetros `default`) para a construção da árvore de classificação e do gráfico correspondente seguem juntamente com os resultados

```
facetree <- ctree(grupo ~ altfac + proffac, data=face)
facetree
```

Model formula:

```
grupo ~ altfac + proffac
```

Fitted party:

```
[1] root
|   [2] altfac <= -0.1: dolico (n = 31, err = 9.7%)
|   [3] altfac > -0.1
|   |   [4] altfac <= 0.8: meso (n = 28, err = 14.3%)
|   |   [5] altfac > 0.8
|   |   |   [6] proffac <= -0.6: meso (n = 17, err = 41.2%)
|   |   |   [7] proffac > -0.6: braq (n = 25, err = 0.0%)
```

```
Number of inner nodes:      3
```

```
Number of terminal nodes: 4
```

```
> plot(facetree)
```

## Árvores–Exemplo 1

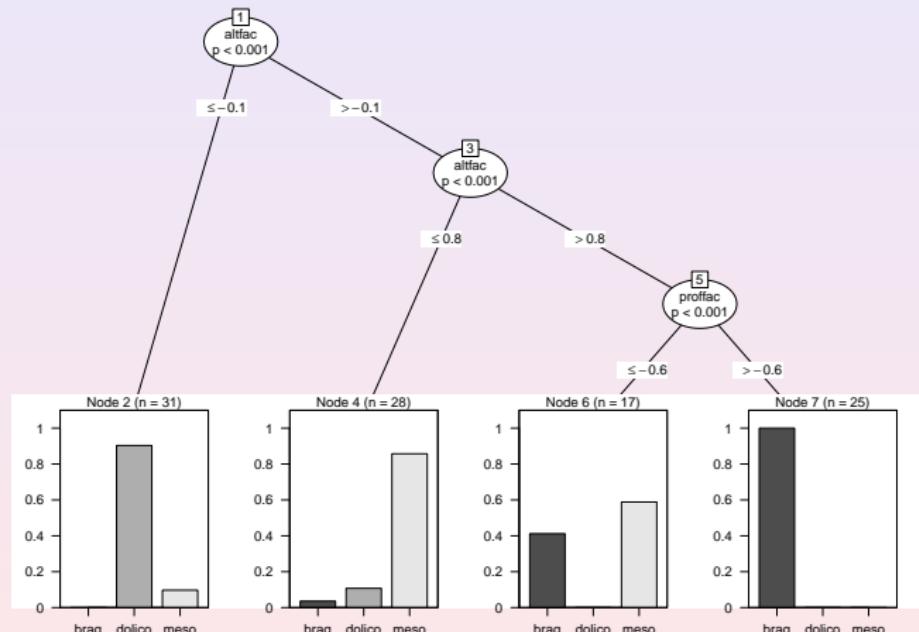


Figura 1: Árvore de decisão para os dados do Exemplo 1.

## Árvores—Exemplo 1

- Na Figura 1, os símbolos ovais, que indicam divisões no espaço das variáveis preditoras são chamados de **nós internos** e os retângulos, que indicam as divisões finais são conhecidos por **nós terminais ou folhas** da árvore.
- Neste exemplo, temos 3 nós internos e 4 nós terminais. Os segmentos que unem os nós são os **galhos** da árvore.
- As barras em cada nó terminal indicam a frequência relativa com que as observações que satisfazem as restrições definidoras de cada galho são classificadas nas categorias da variável resposta.
- As regiões em que o espaço das variáveis preditoras foi particionado estão indicadas na Figura 2.

## Árvores–Exemplo 1

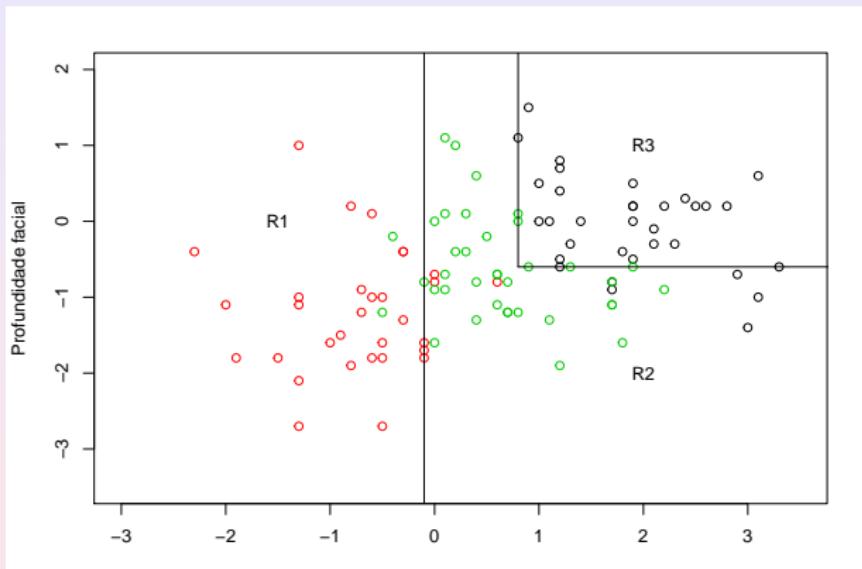


Figura 2: Regiões em que o espaço das variáveis preditoras do Exemplo 1 está particionado (círculos vermelhos = dolicocéfalos, verdes = mesocéfalos, pretos = braquicéfalos).

## Árvores–Exemplo 1

- A variável preditora principal e o correspondente ponto de corte que minimiza a taxa de erros de classificação são **altfac** e  $t = -0,1$ , com  $TEC = 9,7\%$ .
- Para observações com valores  $altfac \leq -0,1$  (região  $R_1$ ), classificamos o indivíduo como dolicocéfalo.
- Para valores de  $altfac > -0,1$ , a classificação depende do valor de **proffac**. Nesse caso, se  $altfac$  estiver entre  $-0,1 \leq 0,8$ , (região  $R_2$ ), classificamos o indivíduo como mesocéfalo com  $TEC = 14,3\%$ .
- Se, por outro lado,  $altfac > 0,8$  e  $proffac \leq -0,6$ , também classificamos o indivíduo como mesocéfalo (região  $R_2$ ), com  $TEC = 41,2\%$ .
- Agora, se  $altfac > 0,8$  e  $proffac > -0,6$ , o indivíduo deve ser classificado como braquicéfalo (região  $R_3$ ), com  $TEC = 0,0\%$ .

## Árvores–Exemplo 1

- Uma tabela com as classificações originais e preditas por meio da árvore de classificação é obtida por meio do comando `predict`

```
table(predict(facetree), face$grupo)
```

	braq	dolico	meso
braq	25	0	0
dolico	0	28	3
meso	8	3	34

- A acurácia do classificador é de 86% e  $TEC = 14\%$ .

## Árvores–Exemplo 2

- Um dos problemas associados à construção de árvores de decisão está relacionado com o **sobreajuste** (*overfitting*).
- Se não impusermos uma regra de parada para a construção dos nós, o processo é de tal forma flexível que o resultado final pode ter tantos nós terminais quantas forem as observações, gerando uma árvore em que cada observação é classificada perfeitamente.
- Para contornar esse problema, pode-se considerar o procedimento conhecido como **poda**, que engloba técnicas para limitar o número de nós terminais das árvores.
- A ideia que fundamenta essas técnicas está na construção de árvores com menos nós e, consequentemente, com menor variância e interpretabilidade. O preço a pagar é um pequeno aumento no viés.

## Árvores–Exemplo 2

- **Exemplo 2.** Consideremos agora os dados do arquivo **coronarias** provenientes de um estudo cujo objetivo era avaliar fatores prognósticos de lesão obstrutiva coronariana (L03) com categorias  $1 : \geq 50\%$  ou  $0 : < 50\%$  em 1500 pacientes.
- Embora tenham sido observadas cerca de 70 variáveis preditoras, aqui trabalharemos com SEXO (0=fem, 1=masc), DIAB (diabetes: 0=não, 1=sim), IMC (índice de massa corpórea), IDADE1 (idade), TRIG (concentração de triglicérides) e GLIC (concentração de glicose).
- Com propósito didático eliminamos casos em que havia dados omissos em alguma dessas variáveis, de forma que 1034 pacientes foram considerados na análise.
- Os comandos do pacote **rpart** para a construção da árvore de classificação por meio de validação cruzada com os resultados correspondentes e o gráfico associado, disposto na Figura 3 seguem:

## Árvores–Exemplo 2

```
lesaoobs <- rpart(formula = L03 ~ IDADE1 + SEXO + GLIC + DIAB +
                     IMC + TRIG, method="class", data = coronarias3,
                     xval = 20, minsplit=10, cp=0.005)
printcp(lesaoobs)
```

Variables actually used in tree construction:

```
[1] GLIC IDADE1 IMC SEXO TRIG
```

Root node error: 331/1034= 0.32012

n= 1034

## Árvores–Exemplo 2

	CP	nsplit	rel error	xerror	xstd
1	0.0453172	0	1.00000	1.00000	0.045321
2	0.0392749	3	0.85801	0.97281	0.044986
3	0.0135952	4	0.81873	0.88218	0.043733
4	0.0090634	6	0.79154	0.87915	0.043687
5	0.0075529	7	0.78248	0.88822	0.043823
6	0.0060423	11	0.75227	0.92749	0.044386
7	0.0050000	13	0.74018	0.97885	0.045062

- A função `rpart()` tem um procedimento de VC embutido, ou seja, usa o CTr para construir a árvore e outro (CTeste) para avaliar a taxa de erros de previsão, repetindo o processo várias vezes. Cada linha da tabela representa um nível diferente da árvore. O erro de classificação obtido por validação cruzada tende a aumentar, pelo menos após o nível ótimo.
- A taxa de erro no **nó raiz** (root node error) corresponde à decisão obtida quando todas as observações do conjunto de dados são classificados na categoria LO3> 50%.
- O erro relativo (**rel error**) mede o erro relativo de classificação dos dados de treinamento obtido por intermédio da árvore. O termo rotulado **xerror** mede o erro relativo de classificação dos elementos do conjunto teste por intermédio da árvore construída com os dados do CTr, e o correspondente erro padrão é rotulado **xstd**.

## Árvores–Exemplo 2

- O produto **root node error**  $\times$  **rel error** corresponde ao valor absoluto da taxa de erros obtida no CTr ( $0,263=0,320 \times 0,740$  para a árvore com 13 subdivisões). O produto **root node error**  $\times$  **xerror** corresponde ao valor absoluto da taxa de erros obtida no CTeste e corresponde a uma medida mais objetiva da acurácia da previsão ( $0,313=0,320 \times 0,979$  para a árvore com 13 subdivisões).
- A árvore gerada obtida por intermédio do comando

```
> rpart.plot(lesaoobs, clip.right.labs = TRUE, under = FALSE,  
             extra = 101, type=4)
```

está disposto na Figura 3.

## Árvores–Exemplo 2

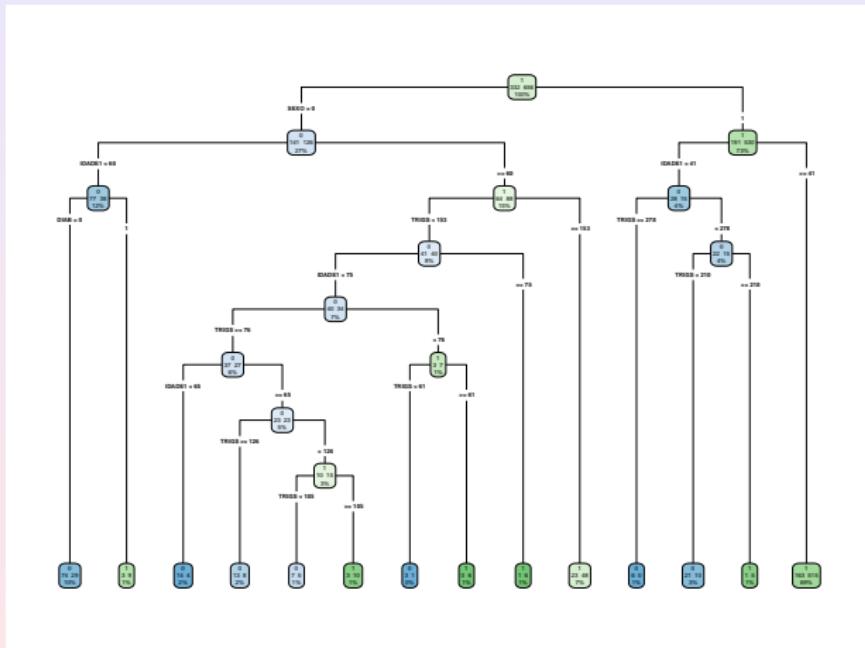


Figura 3: Árvore de decisão para os dados do Exemplo 2.

## Árvores–Exemplo 2

- A tabela com as classificações originais e preditas é obtida por meio do comando

```
table(coronarias\$L03, predict(lesaoobs, type="class"))  
0 1  
0 145 186  
1 59 644
```

e indica um erro de classificação de  $23,4\% = (186 + 59)/1034$ .

- O parâmetro CP (**complexity parameter**) serve para controlar o tamanho da árvore e corresponde ao menor incremento no custo do modelo necessário para a adição de uma nova variável.
- Um gráfico com a variação desse parâmetro com o número de nós, obtido com o comando `plotcp()` pode ser visto na Figura 4.

## Árvores–Exemplo 2

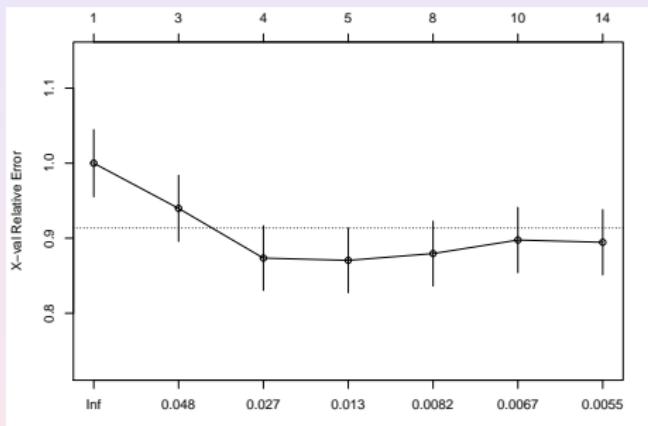


Figura 4: Gráfico CP para o ajuste da árvore aos dados do Exemplo 2.

Nesse gráfico procura-se o nível para o qual o erro relativo obtido por VC é mínimo. Para o exemplo, esse nível é 4, sugerindo que a árvore obtida no exemplo deve ser podada.

## Árvores–Exemplo 3

- Vejamos, agora, ver um exemplo usando o pacote `tree`.
- **Exemplo 3.** Vamos considerar os dados de crianças que foram submetidas a uma cirurgia da coluna para corrigir cifose congênita. Veja Chambers e Hastie (1992). Os dados contém 81 observações, com as variáveis:

$Y$ : cifose: uma variável qualitativa (atributo), com os valores *ausente* e *presente*, indicando se cifose estava ausente ou presente após a cirurgia;

$X_1$ : Age, em meses;

$X_2$ : Number, o número de vértebras envolvidas;

$X_3$ : Start, o número da primeira vértebra (a partir do topo) operada.

- O sumário do ajuste e a Figura 5 estão ilustradas a seguir.

## Árvores–Exemplo 3

Classification tree:

```
tree(formula = Kyphosis ~ Age + Number + Start, data = kyphosis)
Number of terminal nodes: 10
Residual mean deviance: 0.5809 = 41.24 / 71
Misclassification error rate: 0.1235 = 10 / 81
```

- Vemos que a taxa do erro de classificação é 12,35% e o desvio (*deviance*) é definido por

$$-2 \sum_m \sum_{mk} K n_{mk} \log \hat{p}_{mk},$$

onde  $n_{mk}$  é o número de observações no  $m$ -ésimo nó terminal que pertence à  $k$ -ésima classe. Um desvio pequeno indica um bom ajuste aos dados de treinamento. O desvio médio da saída acima é dado pelo desvio dividido por  $n - n_0$ ,  $n_0$  sendo o número de nós terminais, no caso, 10.

- As regiões que determinam a figura são retângulos no espaço tridimensional, logo é complicado, ou não é possível, vê-las em um gráfico.

## Árvores–Exemplo 3

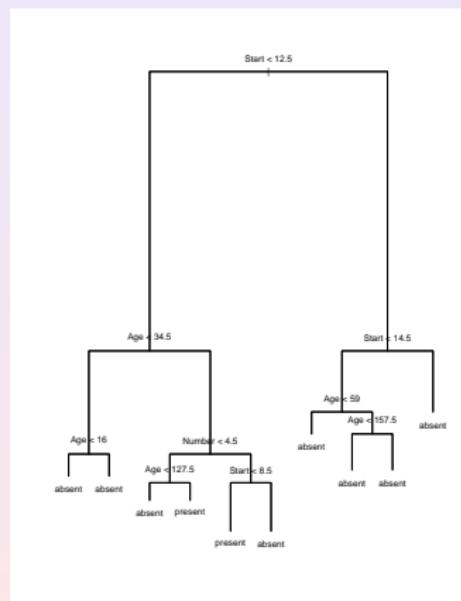


Figura 5: Árvore para o Exemplo 3, com 3 preditores.

## Árvores—Exemplo 3

Para poder ver uma partição, vamos considerar somente dois preditores, Start e Age. A árvore correspondente está na Figura 6 e a partição na Figura 7.

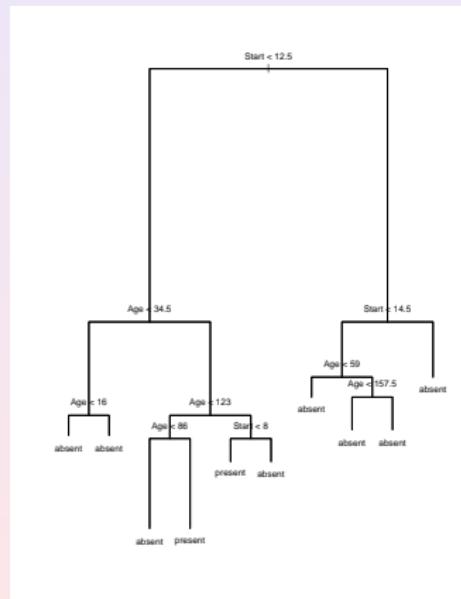


Figura 6: Árvore para o Exemplo 3, com 2 preditores, Start e Age.

## Árvores–Exemplo 3

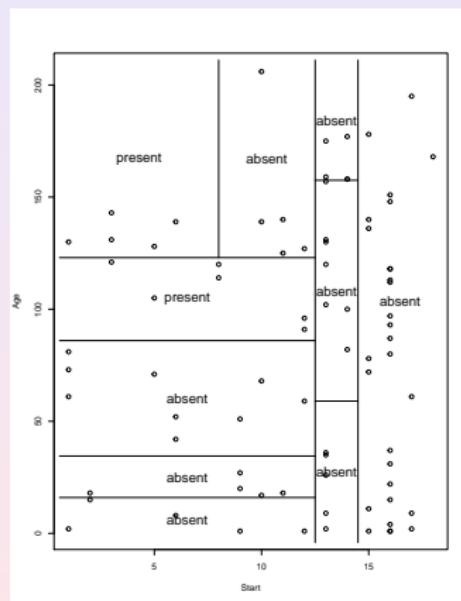


Figura 7: Regiões para o Exemplo 10.3, com 2 preditores.

## Árvores–Exemplo 3

Com relação à Figura 26, algumas regiões que podem ser deduzidas são (sendo  $\mathbf{X} = (X_1, X_2, X_3)'$ ):

$R_1 = \{\mathbf{X} : X_1 < 16, X_3 < 12, 5\}$ , o valor previsto de  $Y$  é ausente;

$R_2 = \{\mathbf{X} : 16 < X_1 < 34, 5, X_3 < 12, 5\}$ , o valor previsto de  $Y$  é ausente;

$R_3 = \{\mathbf{X} : 34, 5 < X_1 < 127, 5, X_2 < 4, 5, X_3 < 12, 5\}$ , o valor previsto de  $Y$  é ausente etc.

## Poda de uma árvore

- No Exemplo 2 (coronarias), vimos que a árvore deve ser podada, inspecionando o parâmetro de complexidade, CP, que indica valores entre 0,011 e 0,023, com nós entre 5 e 7.
- Uma regra empírica para se efetuar a poda da árvore consiste em escolher a **menor árvore** para a qual o valor de **xerror** é menor do que a soma do menor valor observado de **xerror** com seu erro padrão **xstd**. No exemplo em estudo, o menor valor de **xerror** é 0,87915 e o de seu erro padrão **xstd** é 0,043687 de maneira que a árvore a ser construída por meio de poda deverá ser a menor para a qual o valor de **xerror** seja menor que 0,922837 ( $= 0,87915 + 0,043687$ ), ou seja a árvore com 4 subdivisões e 5 nós terminais.
- A poda juntamente com o gráfico da árvore podada e a tabela com os correspondentes valores preditos podem ser obtidos com os comandos a seguir:

## Poda de uma árvore

```
lesaoobspoda <- prune(lesaoobs, cp = 0.015 , "CP", minsplit=20, xval=25)
rpart.plot(lesaoobspoda, clip.right.labs = TRUE, under = FALSE,
           extra = 101, type=4)
rpart.rules(lesaoobspoda, cover = TRUE)

L03                                              cover
0.33 when SEXO is 0 & IDADE1 < 63 & GLIC < 134    13%
0.35 when SEXO is 1 & IDADE1 < 41                      4%
0.59 when SEXO is 0 & IDADE1 >= 63 & GLIC < 134     10%
0.77 when SEXO is 1 & IDADE1 >= 41                     69%
0.85 when SEXO is 0                                     & GLIC >= 134    4%
```

## Poda de uma árvore

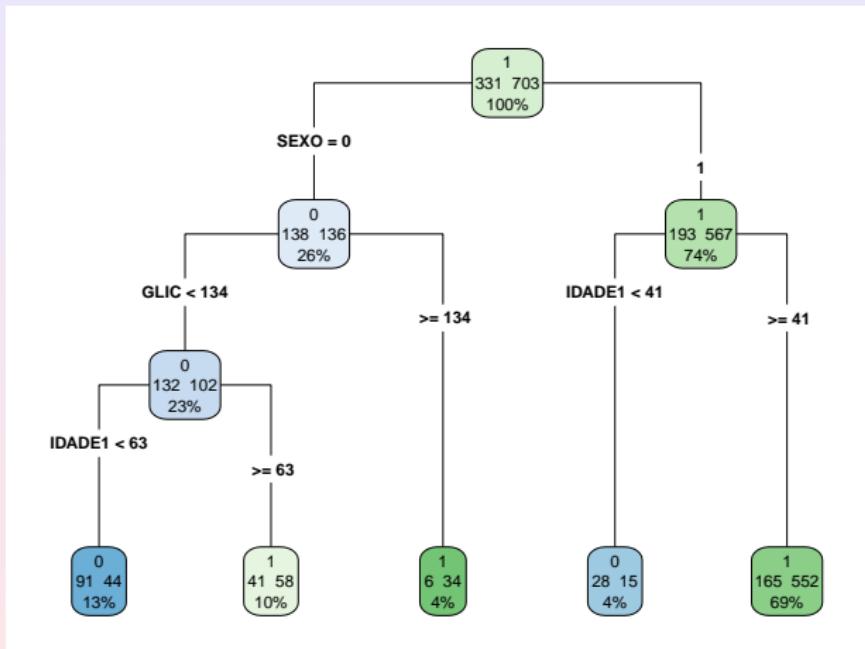


Figura 8: Árvore (podada) ajustada aos dados do Exemplo 2.

## Poda de uma árvore

- O número indicado na parte superior de cada nó da Figura 8 indica a classe majoritária, na qual são classificadas os elementos do conjunto de treinamento; o valor à esquerda no centro do nó representa a frequência desses elementos pertencentes à classe  $LO_3 = 0$  e o valor à direita corresponde à frequência daqueles pertencentes à classe  $LO_3 = 1$ . Na última linha aparece a porcentagem de elementos correspondentes a cada nó terminal.
- A tabela de classificação para o conjunto de dados original (treinamento + validação) obtida a partir da árvore podada é

0	1
0	119 212
1	59 644

Nessa tabela, as linhas indicam a classificação real e as colunas informam a classificação predita pela árvore podada. O erro de classificação, 26,2%, é ligeiramente maior que o erro obtido com a árvore original, bem mais complexa.

## Bagging

- Usualmente, árvores de decisão produzem resultados com grande variância, ou seja, dependendo de como o conjunto de dados é subdividido em conjuntos de treinamento e de teste, as árvores produzidas podem ser diferentes. As técnicas que descreveremos a seguir têm a finalidade de reduzir essa variância.
- A técnica de **agregação bootstrap** (*bootstrap aggregating*) ou, simplesmente **bagging**, é um método para gerar múltiplas versões de um previsor (ou classificador) a partir de vários conjuntos de treinamento e, com base nessas versões, construir um previsor (ou classificador) agregado.
- O ideal seria ter vários conjuntos de treinamento, construir previsores e tomar uma média (no caso de regressão) dos previsores. Todavia, na prática, isso não acontece. Uma maneira de prosseguir, é usar réplicas bootstrap do conjunto de treinamento.

## Bagging

- Suponha que tenhamos o conjunto de treinamento  $(x_1, y_1), \dots, (x_n, y_n)$  e considere o caso de  $y$  quantitativa (regressão). Considere  $B$  réplicas bootstrap desse conjunto e para cada réplica  $b$ , obtemos o previsor de  $y$ ,  $\hat{f}^{*b}(x)$  e construímos o previsor

$$\hat{f}_{\text{bag}} = \frac{1}{B} \sum_{b=1}^B \hat{f}^{*b}(x). \quad (1)$$

- No caso de classificação, para uma observação teste, considere a classe prevista para cada uma das  $B$  árvores e, então, tome como previsor a classe de maior ocorrência comum entre os  $B$  previsores (voto majoritário). Especificamente,

$$\hat{c}_{\text{bag}}(x) = \operatorname{argmax}_k [\#\{(b | \hat{c}^b(x) = k)\}]$$

em que  $\#\{A\}$  denota a cardinalidade do conjunto  $A$ .

- O número de réplicas bootstrap sugerido por Breiman (1996) é cerca de 25. Para detalhes sobre bootstrap veja Efron e Tibshirani (1993) e Notas de Capítulo.

## Bagging—Exemplo 4

- **Exemplo 4.** A técnica bagging pode ser aplicada aos dados do Exemplo 2 por meio dos comandos

```
> set.seed(054)
>
> # train bagged model

> lesaoobsbag <- bagging(
+   formula = L03 ~ SEXO + IDADE1+ GLIC,
+   data = coronarias3,
+   nbagg = 200,
+   coob = TRUE,
+   control = rpart.control(minsplit = 20, cp = 0.015)
+ )
>
> lesaoobspred <- predict(lesaoobsbag, coronarias3)
> table(coronarias3\$L03, predict(lesaoobsbag, type="class"))
```

	0	1
0	117	214
1	78	625

- Variando a semente do processo aleatório, as taxas de erros de classificação giram em torno de 27% a 28%.

## Bagging—Exemplo 4

Quatro das 200 árvores obtidas em cada réplica bootstrap podem ser obtidas por meio dos comandos a seguir e estão representadas na Figura 9.

```
as.data.frame(coronarias4)
clr12 = c("#8dd3c7", "#ffffb3", "#bebada", "#fb8072")
n = nrow(coronarias4)
par(mfrow=c(2,2))
sed=c(1,10,22,345)
for(i in 1:4){
  set.seed(sed[i])
  idx = sample(1:100, size=n, replace=TRUE)
  cart = rpart(L03 ~ DIAB + IDADE1 + SEXO, data=coronarias4[idx,], mod
  prp(cart, type=1, extra=1, box.col=clr12[i])
}
```

## Bagging—Exemplo 4

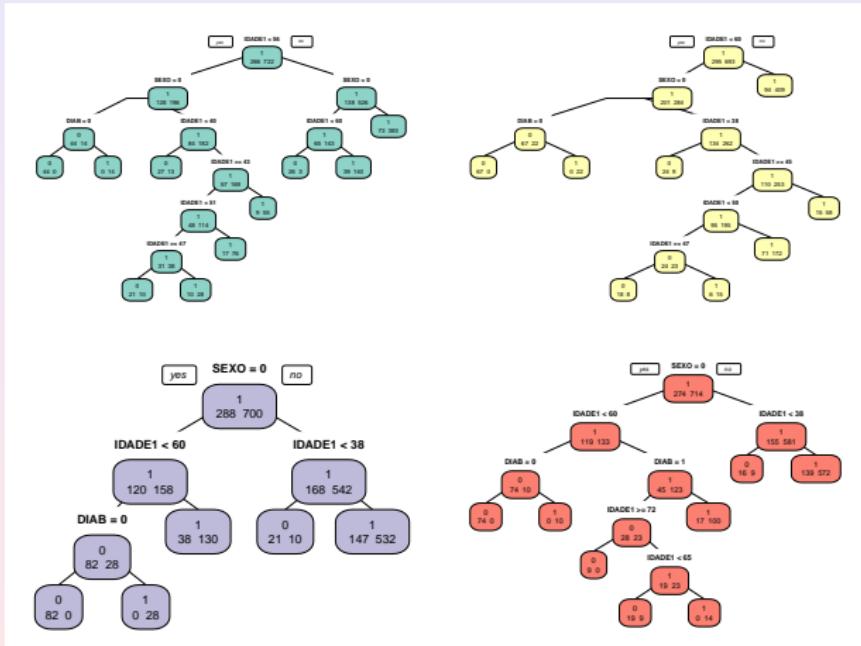


Figura 9: Exemplos de Árvores obtidas por bagging para os dados do Exemplo 2.

## Boosting I

- ① O objetivo do procedimento **boosting** é reduzir o viés e a variância em modelos utilizados para aprendizado supervisionado.
- ② A ideia básica é considerar um conjunto de previsores (**classificadores fracos**) de modo a obter um **previsor (classificador) forte**.
- ③ Classificadores fracos têm taxas de erro de classificação altas. No caso binário, por exemplo, isso corresponde a uma taxa próxima de 0,50, que seria obtida com uma decisão baseada num lançamento de moeda. Um classificador forte, por outro lado, tem uma taxa de erro de classificação baixa.
- ④ Diferentemente da técnica bagging, em que  $B$  árvores são geradas independentemente por meio de bootstrap, com cada observação tendo a mesma probabilidade de ser selecionada em cada um dos conjuntos de treinamento, no procedimento boosting, as  $B$  árvores são geradas **sequencialmente** a partir de um único conjunto de treinamento, com probabilidades de seleção (pesos) diferentes atribuídos às observações.

## Boosting II

- ⑤ Observações mal classificadas em uma árvore recebem pesos maiores para seleção na árvore subsequente (obtida do mesmo conjunto de treinamento), com a finalidade de dirigir a atenção aos casos em que a classificação é mais difícil.
- ⑥ Em ambos os casos, o classificador final é obtido por meio da aplicação dos  $B$  classificadores fracos gerados com as diferentes árvores, por meio do voto majoritário. Além dos pesos atribuídos às observações no processo de geração dos classificadores fracos, o procedimento boosting atribui pesos a cada um deles, em função das taxas de erros de classificação de cada um.
- ⑦ Essencialmente, o classificador forte pode ser expresso como

$$\hat{c}_{\text{boost}}(x) = \sum_{b=1}^B \hat{c}^b(x) w(b) \quad (2)$$

em que  $w(b)$  é o peso atribuído ao classificador  $\hat{c}^b(x)$ .

## Boosting III

- ⑧ Se por um lado, o procedimento bagging raramente reduz o viés quando comparado com aquele obtido com uma única árvore de decisão, por outro, ele tem a característica de evitar o sobreajuste. Essas características são invertidas com o procedimento boosting.
- ⑨ Existem vários algoritmos para a implementação de boosting. O mais usado é o algoritmo conhecido como **AdaBoost** (de *adaptive boosting*), desenvolvido por Freund e Schapire (1997). Dada a dificuldade do processo de otimização de (2), esse algoritmo considera um processo iterativo de otimização que produz bons resultados embora não sejam ótimos.
- ⑩ A ideia é adicionar o melhor classificador fraco numa determinada iteração ao classificador fraco obtido na iteração anterior, ou seja, considerar o processo

$$\hat{c}_{\text{boost}}^b(\mathbf{x}) = \hat{c}_{\text{boost}}^{b-1}(\mathbf{x}) + \hat{c}^b(\mathbf{x})w(b)$$

em que  $\hat{c}_{\text{boost}}^b(\mathbf{x})$  é o classificador com o melhor incremento no desempenho relativamente ao classificador  $\hat{c}_{\text{boost}}^{b-1}(\mathbf{x})$ .

## Boosting IV

- 11 Nesse contexto, a escolha de  $[\hat{c}^b(\mathbf{x}), w(b)]$  deve satisfazer

$$[\hat{c}^b(\mathbf{x}), w(b)] = \operatorname{argmin}_{[c^b(\mathbf{x}), w(b)]} \{E[\hat{c}_{\text{boost}}^{b-1}(\mathbf{x}) + \hat{c}^b(\mathbf{x})w(b)]\}$$

em que  $E[c^b]$  denota o erro de classificação do classificador  $c^b$ .

- 12 Consideremos, por exemplo, um problema de classificação binária baseado num conjunto de treinamento com  $N$  observações. No algoritmo **AdaBoost**, o classificador sempre parte de um único nó (conhecido como **stump**), em que cada observação tem peso  $1/N$ .
- 13 O ajuste por meio desse algoritmo é realizado por meio dos seguintes passos:
- 1) Ajuste o melhor classificador (fraco) com os pesos atuais e repita os passos seguintes para os  $B - 1$  classificadores subsequentes.
  - 2) Calcule o valor do peso a ser atribuído ao classificador fraco corrente a partir de alguma métrica que indique quanto esse classificador contribui para o classificador forte corrente.
  - 3) Atualize o classificador forte adicionando o classificador fraco multiplicado pelo peso calculado no passo anterior.

## Boosting V

- 4) Com esse classificador forte, calcule os pesos atribuídos às observações de forma a indicar quais devem ser o foco da próxima iteração (pesos atribuídos às observações mal classificadas devem ser maiores que pesos atribuídos a observações bem classificadas).
- 14) Os algoritmos para implementação de boosting têm 3 parâmetros:
  - (i) o número de árvores,  $B$ , que pode ser determinado por validação cruzada;
  - (ii) **parâmetro de encolhimento** (*shrinkage*),  $\lambda > 0$ , pequeno, da ordem de 0,01 ou 0,001, que controla a velocidade do aprendizado;
  - (iii) o número de divisões em cada árvore,  $d$ ; como vimos, o AdaBoost, usa  $d = 1$  e, em geral, esse valor funciona bem.
- 15) O pacote **gbm** implementa o boosting e o pacote **adabag** implementa o algoritmo AdaBoost.

## Boosting–Exemplo 5

**Exemplo 5.** Os dados do CD-esteira contêm informações de 16 variáveis, sendo 4 qualitativas (Etiologia, Sexo, Classe funcional NYHA e Classe funcional WEBER) além de 13 variáveis quantitativas [Idade, Altura, Peso, Superfície corporal, Índice de massa corpórea (IMC), Carga, Frequência cardíaca (FC), VO<sub>2</sub> RER, VO<sub>2</sub>/FC, VE/VO<sub>2</sub>,VE/VCO<sub>2</sub>] em diversas fases (REP, LAN, PCR e PICO) de avaliação de um teste de esforço realizado em esteira ergométrica. A descrição completa das variáveis está disponível no arquivo dos dados. Neste exemplo vamos usar as variáveis Carga, Idade, IMC e Peso na fase LAN (limiar anaeróbico) como preditores e VO<sub>2</sub> (consumo de oxigênio, como resposta. Para efeito do exemplo, essas variáveis, com as respectivas unidades de medida, serão denotadas como:

$Y$ : consumo de oxigênio (VO<sub>2</sub>), em  $mL/kg/min$ ;

$X_1$ : carga na esteira, em  $W$  (Watts);

$X_2$ : índice de massa corpórea (IMC), em  $kg/m^2$ .

$X_3$ : Idade, em anos;

$X_4$ : Peso, em  $kg$ .

## Boosting–Exemplo 5

Vamos usar o pacote gbm. Com o comando `summary()` obtemos a influência relativa dos preditores e a figure respectiva, Figura 10.

```
summary(boost.esteira)
var      rel.inf
CARGA   43.46858
IMC     24.22825
Idade   16.84917
Peso    15.45401
```

Vemos que os preditores CARGA e IMC são os mais importantes. Podemos produzir gráficos parciais de dependência para essas duas variáveis (Figura 11). Vemos que o consumo de oxigênio cresce com a CARGA e diminui com o IMC.

## Boosting–Exemplo 5

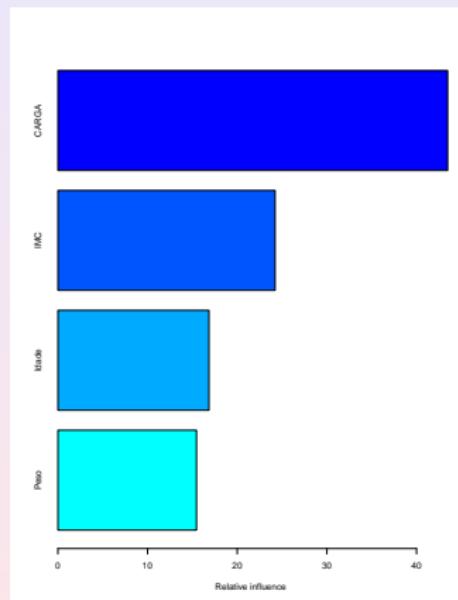


Figura 10: Influência elativa para os preditores do Exemplo 5.

## Boosting–Exemplo 5

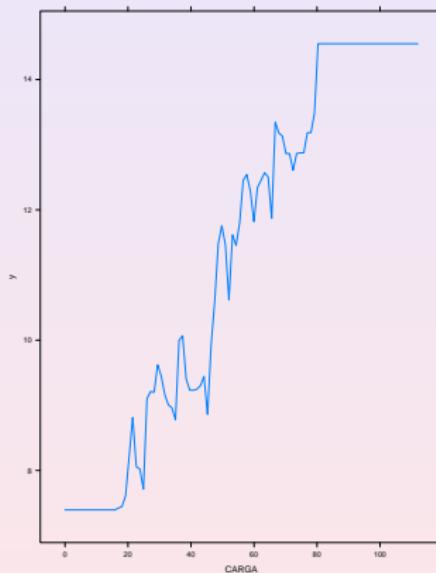


Figura 11: Consumo de oxigênio versus CARGA .

## Boosting–Exemplo 5

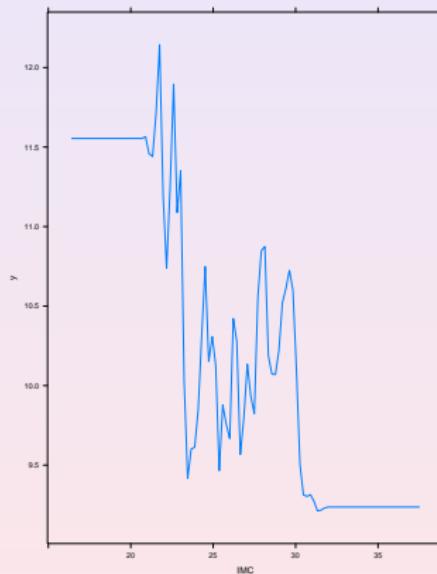


Figura 12: Consumo de oxigênio versus IMC .

## Boosting–Exemplo 5

Agora usamos o modelo via *boosting* para prever VO2 para o conjunto teste.

```
yhat.boost=predict(boost.esteira,newdata=esteira2[-train,],n.trees=5000)
mean((yhat.boost-boston.test)^2)
[1] 2.063152e-05
```

Vemos que o EQM obtido via *boosting* é consideravelmente menor do que aquele obtido via *bagging*.

## Florestas

- Bagging e florestas aleatórias têm o mesmo objetivo: diminuir a variância e o viés de procedimentos baseados em árvores de decisão.
- Enquanto os dois primeiros enfoques são baseados em um conjunto de  $B$  árvores utilizando o mesmo conjunto de  $p$  variáveis preditoras em cada um deles, o enfoque conhecido por **florestas aleatórias** utiliza diferentes conjuntos das variáveis preditoras na construção de cada árvore.
- Na construção de um novo nó, em vez de escolher a melhor variável dentre as  $p$  disponíveis no conjunto de treinamento, o algoritmo de florestas aleatórias seleciona a melhor delas dentre um conjunto de  $m < P$  selecionadas ao acaso. Usualmente, escolhe-se  $m \approx \sqrt{p}$ .

## Florestas

- Formalmente, para cada árvore  $j$ , um vetor aleatório  $\theta_j$  é gerado, independentemente de vetores prévios  $\theta_1, \dots, \theta_{j-1}$ , mas com a mesma distribuição. A árvore usando o conjunto de treinamento e  $\theta_j$  resulta num classificador  $\hat{f}_j(\mathbf{x}, \theta_j)$ . Cada árvore vota na classe mais popular para o vetor  $\mathbf{x}$ .
- A acurácia das árvores aleatórias é tão boa quanto a do *AdaBoost* e, às vezes, melhor. O resultado obtido por intermédio do algoritmo de árvores aleatórias em geral é mais robusto com relação a valores atípicos e ruído além de ser mais rápido do que bagging e boosting.
- O pacote **randomForest** do R implementa florestas. Nesse caso, o número de preditores tem que ser menor do que  $p$ .

## Florestas–Exemplo 6

- Exemplo 6. Vamos usar o conjunto de dados `esteira2` e as variáveis CARGA e IMC para crescer uma floresta. Obtemos o sumário abaixo:

Call:

```
randomForest(formula = V02 ~ Idade + CARGA + IMC + Peso,  
             data = esteira2, mtry = 2, importance = TRUE,  
subset = train)
```

Type of random forest: regression

Number of trees: 500

No. of variables tried at each split: 2

Mean of squared residuals: 4.164505

Percentage Var explained: 52.23

- Obtendo-se previsões para o conjunto teste, temos o sumário abaixo:

```
yhat.rf=predict(rf.esteira,newdata=esteira2[-train,])  
mean((yhat.rf-esteira.test)^2)  
[1] 3.713565
```

- O EQM de previsão via floresta é menor do que aquele via *bagging*, mas maior do que o via *boosting*.

## Florestas–Exemplo 6

Um sumário da importância de cada preditor é dado abaixo:

```
summary(boost.esteira)
var rel.inf
CARGA 43.46858
IMC    24.22825
Idade  16.84917
Peso   15.45401
```

e um gráfico está na Figura 13. Novamente, as variáveis CARGA e IMC são as mais importantes.

Concluindo, sobre os três algoritmos, *bagging*, floresta e *boosting*, o último é o que apresentou o maior poder preditivo.

## Florestas–Exemplo 6

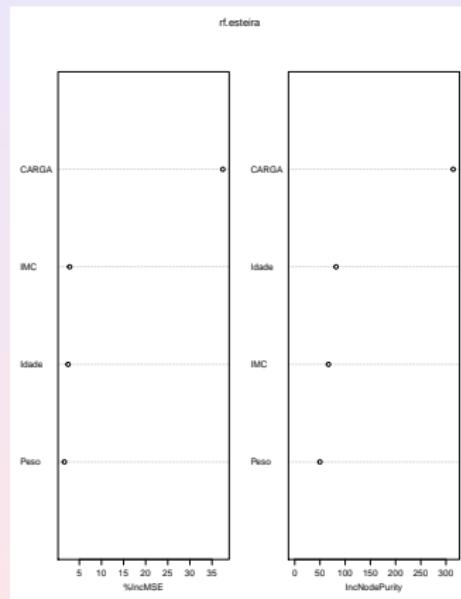


Figura 13: Importância relativa os preditores do Exemplo 6.

## Árvores para regressão

- Consideremos uma situação com variáveis preditoras  $X_1, \dots, X_p$  e uma variável resposta quantitativa  $Y$ . Quando a relação entre variáveis resposta e preditoras for compatível com um modelo linear (no caso de uma relação polinomial, por exemplo), o uso de regressão linear é conveniente e obtemos modelos com interpretabilidade e poder preditivo satisfatórios.
- Quando essa condição não for satisfeita (no caso de modelos não lineares, por exemplo), o uso de árvores pode ser mais apropriado. Além disso, algumas das variáveis preditoras podem ser qualitativas e nesse caso não é necessário transformá-las em **variáveis fictícias** (*dummy variables*) como no caso de modelos de regressão usuais.
- A ideia subjacente é similar àquela empregada em modelos de classificação: subdividir o espaço gerado pelas variáveis explicativas em várias regiões e adotar como previsores, as respostas médias em cada região. As regiões são selecionadas de forma a produzir o menor **erro quadrático médio** ou o menor **coeficiente de determinação**.
- A construção de árvores de regressão pode ser concretizada por meio dos pacotes `tree` e `rpart` entre outros. Ilustraremos a construção de uma árvore para regressão por meio de um exemplo.

## Árvores para regressão - Exemplo 7

- Os dados do arquivo **antracose** foram extraídos de um estudo cuja finalidade era avaliar o efeito da idade (**idade**), tempo vivendo em São Paulo (**tmunic**), horas diárias em trânsito (**htransp**), carga tabágica (**cargatabag**), classificação sócio-econômica (**ses**), densidade de tráfego na região onde o indivíduo morou (**densid**) e distância mínima entre a residência a vias com alta intensidade de tráfego (**distmin**) num índice de antracose (**antracose**) que é uma medida de fuligem (*black carbon*) depositada no pulmão.
- Como esse índice varia entre 0 e 1, consideramos

$$\logrc = \log[\text{índice de antracose}/(1 - \text{índice de antracose})]$$

como variável resposta.

- Inicialmente, construímos uma árvore de regressão para os dados por meio de validação cruzada. Os comandos do pacote **rpart** para gerar a árvore apresentada na Figura 14 são dados a seguir.

## Árvores para regressão - Exemplo

```
pulmaotree2 <- rpart(formula = logrc ~ idade + tmunic + htransp +
    cargatabag + ses + densid + distmin, data=pulmao,
    method="anova", xval = 30, cp=0.010)
rpart.plot(pulmaotree2, clip.right.labs = TRUE, under = FALSE,
            extra = 101, type=4)
```

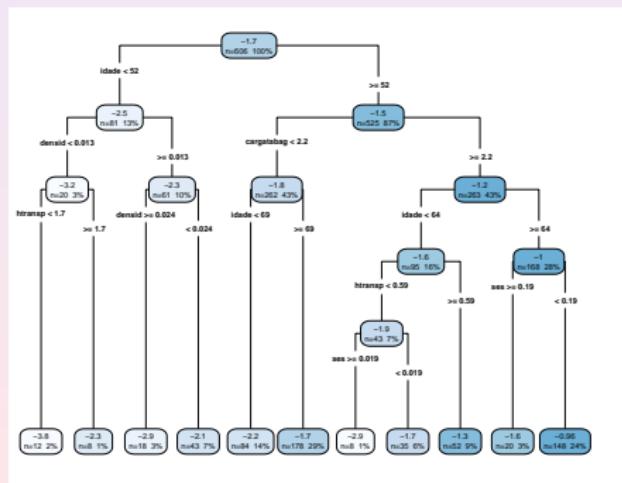


Figura 14: Árvore de decisão para os dados do Exemplo 7.

## Árvores para regressão - Exemplo 7

- Os valores apresentados na parte superior de cada nó são as previsões associadas às regiões correspondentes. Na parte inferior encontram-se o número e a porcentagem de elementos incluídos em cada região.
- Para evitar um possível sobreajuste da árvore proposta, convém avaliar o efeito de seu tamanho segundo algum critério. No pacote **rpart** esse critério é baseado no parâmetro de complexidade (*CP*) que está relacionado com o número de nós terminais e na relação  $1 - R^2$  em que  $R^2$  tem a mesma interpretação do coeficiente de determinação utilizado em modelos lineares.
- Com essa finalidade, podemos utilizar o comando **rsq.rpart(pulmaotree2)**, que gera a tabela com os valores de *CP* e os gráficos apresentados na Figura 15.

## Árvores para regressão - Exemplo 7

Variables actually used in tree construction:

```
[1] cargatabag densid      htransp      idade      ses
```

Root node error: 765.87/606 = 1.2638

n= 606

	CP	nsplit	rel error	xerror	xstd
1	0.087230	0	1.00000	1.00279	0.077419
2	0.062020	1	0.91277	0.96678	0.076624
3	0.025220	2	0.85075	0.91078	0.072406
4	0.024106	3	0.82553	0.91508	0.073489
5	0.015890	4	0.80142	0.85814	0.069326
6	0.014569	5	0.78553	0.87882	0.070312
7	0.011698	6	0.77097	0.89676	0.070309
8	0.011667	7	0.75927	0.90730	0.069025
9	0.011347	8	0.74760	0.91268	0.069066
10	0.010289	9	0.73625	0.91536	0.069189
11	0.010000	10	0.72596	0.91250	0.069202

## Árvores para regressão - Exemplo 7

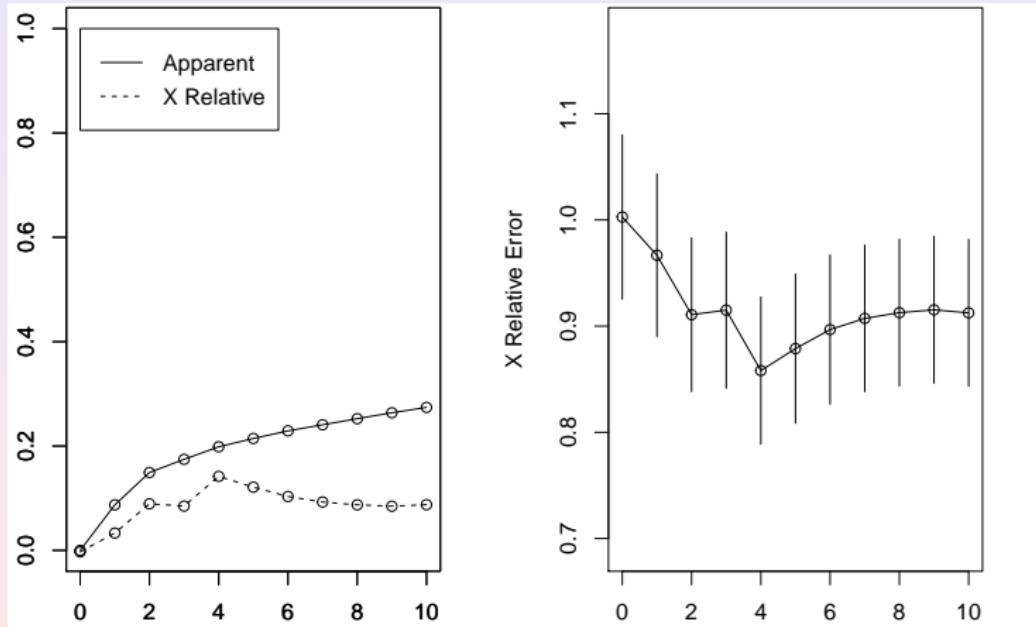


Figura 15: Efeito do número de divisões para a árvore ajustada aos dados do Exemplo 7.

## Árvores para regressão - Exemplo 7

Utilizando o mesmo critério aplicado no caso de classificação, a sugestão é podar a árvore, fixando o número de subdivisões em 4, correspondendo a 5 nós terminais.

O gráfico da árvore podada (disposto na Figura 16), além da regras de partição do espaço das variáveis explicativas podem ser obtidos com os comandos

```
cpmin <- pulmaotree2$cptable[which.min(pulmaotree2$cptable[, "xerror"]),
                                "CP"]
pulmaotree2poda <- prune(pulmaotree2, cp = cpmin)
rpart.plot(pulmaotree2poda, clip.right.labs = TRUE, under = FALSE,
            extra = 101, type=4)
pulmaotree2poda$cptable
      CP nsplit rel error      xerror      xstd
1 0.08722980      0 1.0000000 1.0027880 0.07741860
2 0.06201953      1 0.9127702 0.9667786 0.07662398
3 0.02521977      2 0.8507507 0.9107847 0.07240566
4 0.02410635      3 0.8255309 0.9150809 0.07348898
5 0.01588985      4 0.8014246 0.8581417 0.06932582
> rpart.rules(pulmaotree2poda, cover = TRUE)
logrc                      cover
-2.5 when idade < 52          13%
-2.2 when idade is 52 to 69 & cargatabag < 2.2  14%
-1.7 when idade >= 69 & cargatabag < 2.2      29%
-1.6 when idade is 52 to 64 & cargatabag >= 2.2  16%
-1.0 when idade >= 64 & cargatabag >= 2.2      28%
```

## Árvores para regressão - Exemplo 7

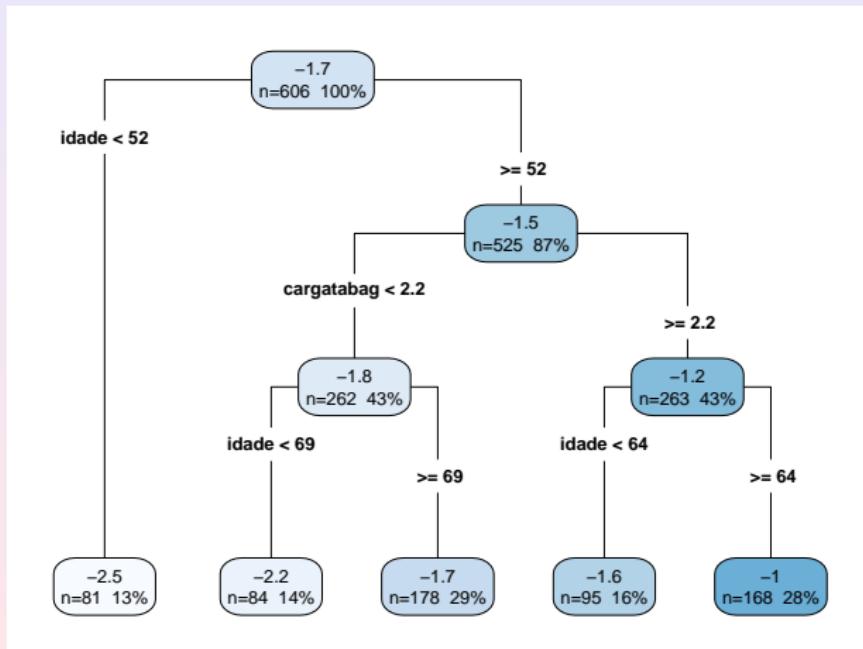


Figura 16: Árvore podada ajustada aos dados do Exemplo.

## Árvores para regressão - Exemplo 7

Valores preditos para o conjunto de dados com o respectivo *RMSE* são obtidos por meio dos comandos

```
rmse = function(actual, predicted) {  
  sqrt(mean((actual - predicted)^2))  
}  
  
rmse(predpulmatree2poda, pulmao$logrc)  
[1] 1.006402
```

Convém notar que embora a utilização de árvores de decisão gere um modelo bem mais simples do que aquele obtido por meio de análise regressão, os objetivos são bem diferentes.

Nesse contexto, o modelo baseado em árvores deve ser utilizado apenas quando o objetivo é fazer previsões, pois pode deixar de incluir variáveis importantes para propósitos inferenciais. No Exemplo, uma das variáveis mais importantes para entender o processo de deposição de fuligem no pulmão é o numero de horas gastas em trânsito. Essa variável foi incluída significativamente no ajuste do modelo de regressão mas não o foi no modelo baseado em árvores.

Há um algoritmo **boosting** aplicável na construção de árvores para regressão. Veja Hastie et al. (2017, cap. 10) para detalhes.

## Referências

- Breiman, L. (2001). Random forests. *Machine Learning*, **45**, 5–32.
- Chambers, J. M and Hastie, T. J. (1992:). *Statistical Models in S*. California: Wadsworth and Brooks/Cole.
- Hastie, T. J. and Tibshirani, R. J. (1990). *Generalized Additive Models*. London: Chapman and Hall.
- James, G., Witten, D., Hastie, T. and Tibshirani, R. (2017). *Introduction to Statistical Learning*. Springer.
- Morettin, P. A. e Singer, J. M. (2022). *Estatística e Ciência de Dados*. LTC, Rio de Janeiro.

## MAE 5905: Introdução à Ciência de Dados

Pedro A. Morettin

Instituto de Matemática e Estatística  
Universidade de São Paulo  
[pam@ime.usp.br](mailto:pam@ime.usp.br)  
<http://www.ime.usp.br/~pam>

### Aula 12

18 de maio de 2023

# Sumário

- 1 Análise de Agrupamentos
- 2 Estratégias de agrupamento
- 3 Algoritmos hierárquicos
- 4 Algoritmos de partição: K-médias
- 5 AA-Tópicos Adicionais
- 6 Outras distâncias

## Preliminares

- Na Análise de Agrupamentos (AA), o objetivo é agrupar “pontos” pertencentes a determinado espaço em “grupos” de acordo com alguma medida de distância, de modo que pontos num mesmo grupo tenham uma pequena distância entre eles.
- A AA é, às vezes, chamada de segmentação de dados. O espaço mencionado acima pode ser um espaço Euclidiano, eventualmente de dimensão grande, ou pode ser um espaço não Euclidiano, por exemplo quando queremos agrupar documentos segundo certos tópicos.
- O problema da AA insere-se naquilo que chamamos anteriormente de aprendizado não supervisionado, ou seja, temos apenas um conjunto de variáveis preditoras (inputs), não há uma variável resposta, e o objetivo é descrever associações e padrões entre essas variáveis.
- Para alcançar nosso objetivo, podemos usar várias técnicas de agrupamento, e nesta aula iremos discutir algumas delas.

## Preliminares

A AA é usada em muitas áreas e aplicações, como:

- i) segmentação de imagens como em fMRI (imagens por ressonância magnética funcional), em que se pretende particionar a imagem em áreas de interesse. Veja, por exemplo, Sato et al. (2007);
- ii) bioinformática, por exemplo na análise de expressão de genes gerados de *microarrays* ou sequenciamento de DNA ou proteínas. Veja Hastie et al. (2009) ou Fujita et al. (2007), por exemplo.
- iii) reconhecimento de padrões, de objetos e caracteres, por exemplo identificação de textos escritos a mão em linguística;
- iv) redução (ou compressão) de grandes conjuntos de dados, a fim de escolher grupos de dados de interesse.

## AA-Exemplo 1

**Exemplo 1.** Consideremos as medidas das variáveis altura ( $X_1$ , em cm), peso ( $X_2$ , em kg), idade ( $X_3$ , em anos) e sexo ( $X_4$ , M ou F) em 12 indivíduos dispostas na Tabela 1.

Tabela 1: Dados de 12 indivíduos

Ind.	Altura	Peso	Idade	Sexo	Alt.Padr.	Peso Padr.	Id. Padr.
A	180	75	30	M	0,53	0,72	0,38
B	170	70	28	F	-0,57	-0,13	-0,02
C	165	65	20	F	-1,12	-0,97	-1,61
D	175	72	25	M	-0,02	0,21	-0,61
E	190	78	28	M	1,63	1,23	-0,02
F	185	78	30	M	1,08	1,23	0,38
G	160	62	28	F	-1,67	-1,48	-0,02
H	170	65	19	F	-0,57	-0,97	-1,81
I	175	68	27	M	-0,02	-0,47	-0,22
J	180	78	35	M	0,53	1,23	1,38
K	185	74	35	M	1,08	0,55	1,38
L	167	64	32	F	-0,90	-1,14	0,78
$\mu$	175,17	70,75	28,08	—	0	0	0
$\sigma$	9,11	5,91	5,02	—	1	1	1

## AA-Exemplo 1

- Nessa tabela, a média de cada variável é indicada por  $\mu$  e o desvio padrão por  $\sigma$ . As colunas 6, 7 e 8 trazem as variáveis padronizadas, ou seja,

$$Z_i = \frac{X_i - \mu_{X_i}}{\sigma_i}, \quad (1)$$

para  $i=1,2,3$

- Para a variável  $X_4$  (sexo), esta padronização, é claro, não faz sentido. A Figura 1 apresenta um gráfico de dispersão de  $X_3$  versus  $X_2$ .

## AA-Exemplo 1

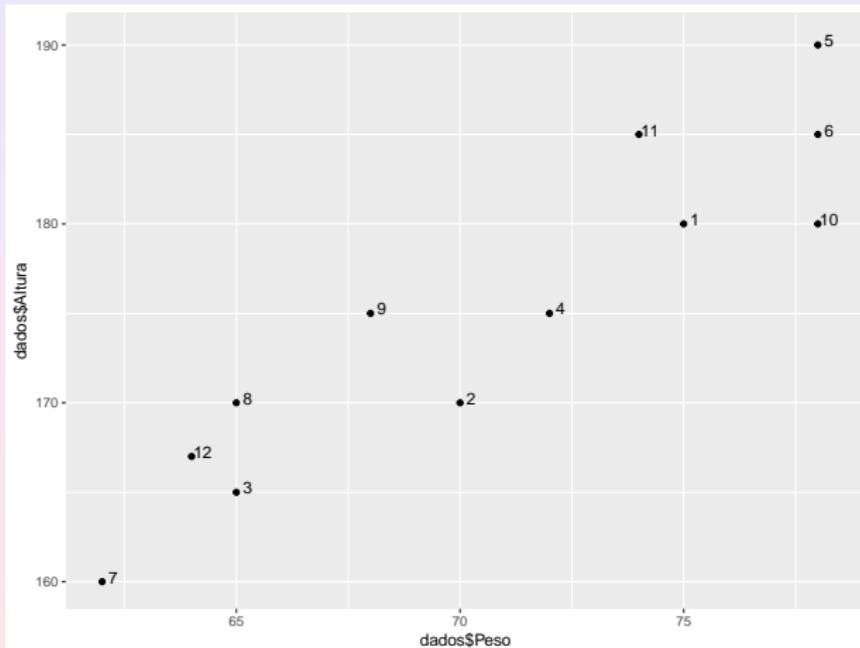


Figura 1: Gráfico de Altura *versus* Peso para os dados da Tabela 1, A=1, B=2 etc.

## AA-Exemplo 1

- O objetivo é agrupar pontos que estejam próximos. Para isso, definamos a **distância Euclidiana** entre dois pontos  $\mathbf{x} = (x_1, x_2)$  e  $\mathbf{y} = (y_1, y_2)$  como

$$d(\mathbf{x}, \mathbf{y}) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2}.$$

- Nesse caso, a dimensão do espaço é dois. No caso geral de um espaço Euclidiano  $p$ -dimensional a distância é definida por

$$d(\mathbf{x}, \mathbf{y}) = \sqrt{(x_1 - y_1)^2 + \dots + (x_p - y_p)^2}.$$

- Além da distância Euclidiana, podemos usar outras distâncias, como

$$d_1 = |x_1 - y_1| + \dots + |x_p - y_p| : \text{distância } L_1 \text{ ou Manhattan}, \quad (2)$$

$$d_2 = \max_{1 \leq i \leq p} |x_i - y_i| : \text{distância } L_\infty. \quad (3)$$

- Para espaços não Euclidianos, há outras definições de distância, como **Hamming**, **cosseno**, **Jaccard**, **edit** etc. (Ver Notas de Capítulo).

## AA–Exemplo 1

- Podemos obter a **matriz de similaridade** dos dados, segundo dada distância.
- Consideremos  $n$  indivíduos, para os quais observamos  $p$  variáveis. A matriz de similaridade será uma matriz  $n \times n$ ,  $D = [d(i,j)]_{i,j=1}^n$ , simétrica, com zeros na diagonal principal (Pode-se considerar uma matriz reduzida, de ordem  $(n - 1) \times n$ , para evitar os zeros).
- Na Tabela 2 temos essa matriz para os dados da Tabela 5, usando a distância Euclidiana. Algumas distâncias em ordem crescente são:

$$\begin{aligned}d(C, L) &= 2,24, \\d(A, J) &= 3,00, \\d(H, L) &= 3,16, \\d(D, I) &= 4,00, \\d(F, K) &= 4,00, \\d(B, H) &= 5,00, \\d(C, H) &= 5,00, \\d(E, F) &= 5,00, \\d(F, J) &= 5,00, \\d(A, K) &= 5,10.\end{aligned}$$

## AA-Exemplo 1

Essas distâncias (indicadas em negrito) nos dão uma ideia de como agrupar pontos.

Tabela 2: Matriz de similaridade, distância Euclidiana

	A	B	C	D	E	F	G	H	I	J	K
B	11,18										
C	18,03	7,07									
D	5,83	5,38	12,21								
E	10,44	21,54	28,18	16,16							
F	5,83	17,00	23,85	11,66	5,00						
G	23,85	12,81	5,83	18,03	34,00	29,68					
H	14,14	5,00	5,00	8,60	23,85	19,85	10,44				
I	8,60	5,39	10,44	4,00	18,03	14,14	16,16	5,83			
J	3,00	12,81	19,85	7,81	10,00	5,00	25,61	16,40	11,18		
K	5,10	15,52	21,93	10,20	6,40	4,00	27,73	17,49	11,66	6,40	
L	17,03	6,71	2,24	11,31	26, 93	22,80	7,28	3,16	8,94	19,10	20,59

## Tipos de algoritmos

Podemos dividir os algoritmos de agrupamento em três grupos (Hastie et al., 2009):

- a) **combinatórios**: trabalham diretamente com os dados, não havendo referência à sua distribuição;
- b) **baseados em modelos**: supõem que os dados sejam uma amostra aleatória simples de uma população, com uma densidade de probabilidade, que é uma mistura de densidades componentes, cada uma descrevendo um grupo;
- c) **bump hunters** : tratam de estimar, de modo não paramétrico, as modas da densidade. As observações mais próximas a cada moda definem os grupos.

## Tipos de algoritmos

Por sua vez, os algoritmos combinatórios podem ser classificados em dois grupos:

- i) **algoritmos hierárquicos**: que ainda podem ser subdivididos em **aglomerativos** e **divisivos**. No primeiro caso, iniciamos o procedimento de modo que cada ponto forma um grupo e vamos combinando grupos com base em suas proximidades, usando alguma definição de proximidade (como uma distância). Paramos quando fixamos um número de grupos ou obtemos grupos que não são desejáveis, por alguma razão. No segundo caso, partimos de um único grupo e por divisões sucessivas obtemos 1,2 etc. grupos. Neste texto, usaremos o método aglomerativo.
- ii) **algoritmos de partição** ( ou obtidos por associação de pontos): os grupos obtidos formam uma partição do conjunto total de pontos. Os pontos são considerados em alguma ordem e cada um deles é associado ao grupo no qual ele melhor se ajusta. O método chamado de **K-médias** pertence a esse grupo de algoritmos.

## Tipos de algoritmos

- Se estivermos num espaço Euclíadiano, quando usamos alguma distância entre os pontos, grupos podem ser caracterizados pelo seu **centróide** (média das coordenadas dos pontos).
- Num espaço não Euclíadiano, não existe a noção de centróide, e deveremos usar alguma outra maneira de caracterizar os grupos.
- Se o conjunto de pontos for muito grande, há o que se chama de **maldição da dimensionalidade** (*curse of dimensionality*). Em grandes dimensões, **quase** todos os pares de pontos têm a mesma distância entre si e quaisquer dois vetores são **quase** sempre ortogonais.

## AH–Exemplo 1

- Para ilustrar o método, vamos voltar ao Exemplo 1 e procuremos os pontos mais próximos, baseando-nos na Figura 1.
- Consideremos as variáveis  $X_1$  (altura) e  $X_2$  (peso) e o gráfico da Figura 1. Temos um espaço Euclidiano de dimensão 2 e a proximidade entre dois pontos medida pela distância Euclidiana (DE).
- Cada grupo será representado pelo seu centróide e a regra de agrupamento será, pois: calcular a distância Euclidiana entre os centróides de dois grupos quaisquer e escolher, para agrupar, os dois grupos com a menor DE.

## AH-Exemplo 1

- i) Inicialmente, cada ponto é considerado um grupo, que coincide com seu centróide.
- ii) Consultando a Tabela 2,  $C=(65, 165)$  e  $L=(64; 167, 64)$  são os mais próximos com DE  $d(C, L) = \sqrt{5,00} = 2,24$  e centróide  $c(C, L) = (64, 5; 166)$ . Esses dois pontos formam o primeiro agrupamento,  $\mathcal{G}_1 = \{C, L\}$ . A distância entre esses pontos, 2,24, será chamada **nível do agrupamento ou junção**.
- iii) A seguir, teríamos que recalcular as distâncias entre os pontos, considerando agora os grupos A, B, CL, D, E, F, G, H, I, J, K. A maioria das distâncias não muda, mudando somente as distâncias dos pontos ao centróide de C e L. Pela Tabela 2, a DE de  $A = (75, 180)$  a  $J = (78, 180)$  é 3, logo agrupamos A e J, formando o grupo  $\mathcal{G}_2 = \{A, J\}$ , com centróide  $c(\mathcal{G}_2) = (76, 5, 180)$  e nível 3,00.
- iv) Agora, DE entre  $D = (72, 175)$  e  $I = (68, 175)$  é 4,00, obtendo-se o agrupamento,  $\mathcal{G}_3 = \{D, I\}$ , com centróide  $c(\mathcal{G}_3) = (70, 175)$  e nível 4,00.

## AH–Exemplo 1

- v) A seguir, os pontos mais próximos são  $F = (78, 185)$  e  $K = (74, 185)$ , com DE igual a 4,00, obtendo-se  $\mathcal{G}_4 = \{F, K\}$ , centróide  $c(\mathcal{G}_4) = (76, 185)$  e nível 4,00.
- vi) Consideramos o ponto  $H = (65, 170)$ , cuja DE ao centróide de C e L é  $d(H, c(\mathcal{G}_1)) \approx 4,03$ . Portanto, obtemos outro agrupamento,  $\mathcal{G}_5 = \{C, H, L\}$ , com centróide  $c(\mathcal{G}_5) = (64, 7; 167, 3)$  e nível 4,03.
- vii) A seguir, agrupamos  $B = (70, 170)$  com  $\mathcal{G}_3 = \{D, I\}$ , notando que a DE entre B e o centróide desse grupo é 5,00, passando a ser esse valor o nível da junção. O novo grupo é  $\mathcal{G}_6 = \{B, D, I\}$ , com centróide  $c(\mathcal{G}_6) = (70; 173, 3)$ .
- viii) Agrupamos, agora, os grupos  $\mathcal{G}_2 = \{A, J\}$  e  $\mathcal{G}_4 = \{F, K\}$ , com DE entre seus centróides de 5,02, formando-se o grupo  $\mathcal{G}_7 = \{A, F, J, K\}$ , com centróide  $c(\mathcal{G}_7) = (76, 25; 182, 5)$  e nível 5,02.

## AH–Exemplo 1

- ix) Agregamos, a seguir, o ponto  $G = (62, 160)$  ao grupo  $\mathcal{G}_5 = \{C, H, L\}$ , obtendo-se o novo grupo  $\mathcal{G}_8 = \{C, G, H, L\}$ , com centróide  $c(\mathcal{G}_8) = (64; 165, 5)$  e nível 7,31.
- x) Finalmente, agrupamos o ponto  $E = (78, 190)$  ao grupo  $\mathcal{G}_7 = \{A, F, J, K\}$ , obtendo-se o grupo  $\mathcal{G}_9 = \{A, E, F, J, K\}$ , com centróide  $c(\mathcal{G}_9) = (76, 6; 184)$  e nível 7,50.

A Figura 2 mostra esses agrupamentos. Podemos prosseguir, agrupando-se dois desses grupos (aqueles que possuem a menor DE entre os respectivos centróides) e, finalmente, agrupar os dois grupos restantes.

## AH-Exemplo 1

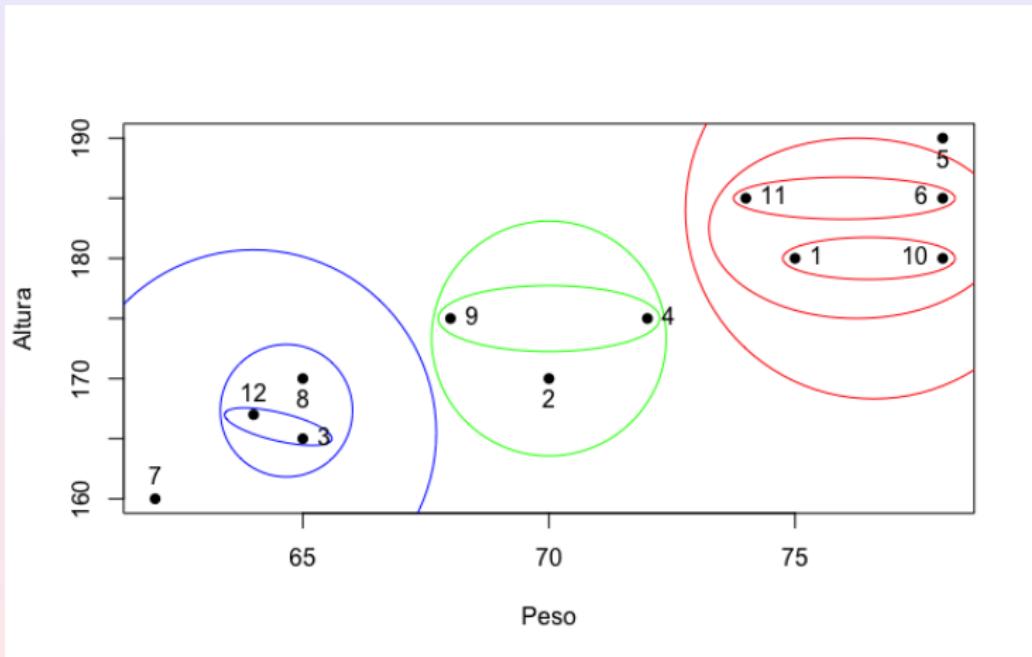


Figura 2: Agrupamentos obtidos para o Exemplo 1.

## AH-Dendrograma

- Um gráfico que sumariza o procedimento ([dendrograma](#)) está na Figura 3. No eixo vertical da figura colocamos os níveis, no horizontal, os pontos de modo conveniente. Nessa figura, usamos a distância Euclidiana.
- Na Tabela 3 temos um resumo do método hierárquico, usando distância Euclidiana e centróides, para agrupar os pontos, para o Exemplo 1.

Tabela 3: Resumo do procedimento de agrupamento para o Exemplo 1

Passo	Agrupamento	Nível
1	C, L	2,24
2	A, J	3,00
3	D, I	4,00
4	F, K	4,00
5	H, CL	4,03
6	B, DI	5,00
7	AJ, KF	5,02
8	G, CHL	7,31
9	E, AFJK	7,50

## AH-Dendrograma

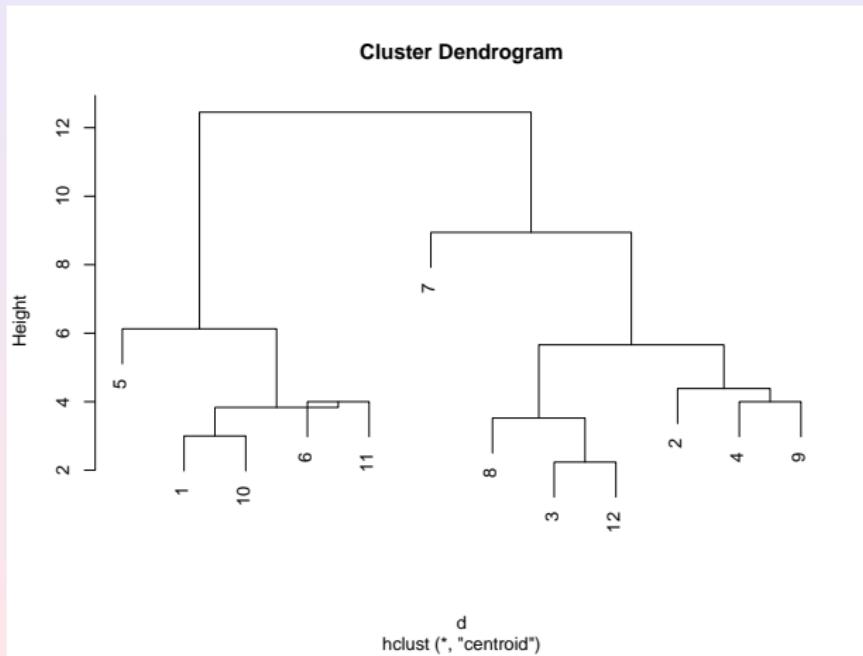


Figura 3: Dendrograma para o Exemplo 1.

## AH–Interpretação

- Interpretando o resultado, com esses três grupos, vemos que o primeiro contém as pessoas menos pesadas e mais baixas (4 pessoas), depois aquele que contém pessoas com pesos e alturas intermediárias (3 pessoas) e, finalmente, o grupo que contém as pessoas mais pesadas e mais altas (5 pessoas).
- Se esse for objetivo, podemos parar aqui. Se o objetivo é obter dois grupos, um com pessoas mais baixas e menos pesadas e, outro, com pessoas mais altas e mais pesadas, continuamos a agrupar mais uma vez, obtendo os grupos  $\mathcal{G}_{10} = \{B, C, D, G, H, I, L\}$  e  $\mathcal{G}_9$ .
- Um dos objetivos da construção de grupos é **classificar** um novo indivíduo em algum dos grupos obtidos. O problema da classificação está intimamente ligado ao problema de AA, e já foi tratado em capítulos anteriores.
- Um pacote do repositório R que pode ser usado é o **cluster**. Após carregar o pacote em sua área por meio de `library(cluster)`, temos que informar a distância (`euclidian`, `maximum`, `manhattan` etc.) a usar e o método de agrupamento (`centroid`, `average`, `median` etc.). O pacote contém várias funções para mostrar em que grupo estão as unidades, obter o dendrograma, fornecer a ordem para fazer o dendrograma etc.

## K-médias

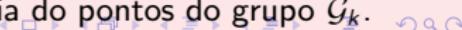
- O método de K-médias tem por objetivo partitionar os pontos em  $K$  grupos, de tal modo que a soma dos quadrados das distâncias dos pontos aos centros dos agrupamentos (**clusters**) seja minimizada. É um método baseado em centróides, como vimos anteriormente, e pertence à classe de algoritmos (ii) discutida antes, e requer que o espaço seja Euclidiano.
- Usualmente, o valor de  $K$  é conhecido e deve ser fornecido pelo usuário mas é possível obtê-lo por tentativa e erro.

O algoritmo mais comum é devido a Hartigan and Wong (1979) é usado como *default* em pacotes computacionais. Outros algoritmos são os de MacQueen (1967), Lloyd (1957) eForgy (1965).

- A função **kmeans** do pacote **cluster** pode ser utilizada.
- A ideia básica consiste em definir grupos em que a variação interna seja minimizada. Esta é, em geral, definida como a soma das DE ao quadrado entre pontos e o centróide correspondente:

$$W(\mathcal{G}_k) = \sum_{x_i \in \mathcal{G}_k} (x_i - \mu_k)^2, \quad (4)$$

em que  $x_i$  é um ponto no grupo  $\mathcal{G}_k$  e  $\mu_k$  é a média do pontos do grupo  $\mathcal{G}_k$ .



## K-médias

O algoritmo consiste nos seguintes passos:

- i) especifique K e selecione K pontos que pareçam estar em diferentes grupos;
- ii) considere esses pontos como os centróides iniciais desses grupos;
- iii) associe cada observação ao centróide mais próximo, baseado na distância Euclidiana entre essa e o centróide;
- iv) para cada um dos K grupos, recalcule o centróide após cada ponto ser incluído.
- v) iterativamente, minimize a soma total de quadrados dentro dos grupos, até que os centróides não mudem muito (o R usa 10 iterações como *default*). A soma total de quadrados dentro dos grupos é definida por

$$\sum_{j=1}^K W(\mathcal{G}_j) = \sum_{j=1}^K \sum_{x_i \in \mathcal{G}_j} (x_i - \mu_j)^2. \quad (5)$$

## K-médias: Exemplo 2

**Exemplo 2.** Consideremos 100 simulações de duas variáveis com distribuição normal, uma com média 0 e desvio padrão 0,3 e outra, com média 1 e desvio padrão também 0,3. Na Figura 4 apresentamos os dois grupos com os respectivos centros, resultantes da aplicação do algoritmo **K-means** com  $K = 2$ .

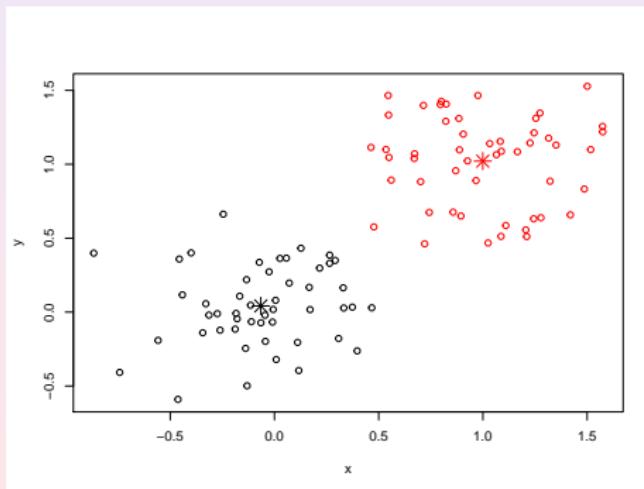


Figura 4: Uso do pacote kmeans para o exemplo simulado

## K-médias: Exemplo 3

- **Exemplo 3.**: Consideremos os dados da Tabela 1 e as variáveis Peso e Altura. Usando o resultado do procedimento hierárquico, suponha que tenhamos  $K = 3$  grupos.
- Usando a função `kmeans`, obtemos o gráfico da Figura 5. Nessa figura temos os três grupos (com tamanhos 3,4 e 5) em diferentes cores e os centros de cada grupo.
- Esses centróides são dados pelo programa,  $(63, 67; 164)$ ,  $(68, 75; 172, 5)$  e  $(76, 60; 184, 0)$ . As somas de quadrados dentro dos grupos são 30,67, 51,75 e 85,20, respectivamente. A soma de quadrados total entre grupos é 87,1.

## K-médias: Exemplo 3

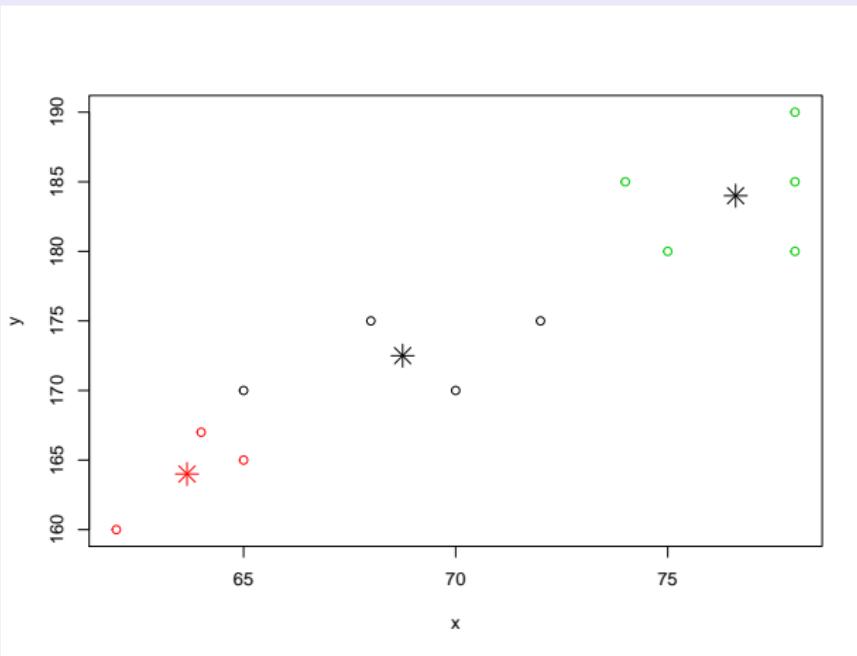


Figura 5: Uso do pacote `kmeans` para o Exemplo 3.

## K-médias: Exemplo 4

- **Exemplo 4:** Vamos considerar os dados de um conjunto de 4.000 motoristas encarregados de fazer entregas de determinados produtos. Há várias variáveis envolvidas, mas iremos considerar somente duas, nomeadamente,  $X_1$ : distância média percorrida por cada motorista (em milhas) e  $X_2$ : porcentagem média do tempo em que o motorista esteve acima do limite de velocidade por mais de 5 milhas por hora. Há dados do setor urbano e rural.
- Os dados podem ser obtidos de  
[https://raw.githubusercontent.com/datasets/introduction-to-k-means-Clustering/master/Data/data\\_1024.csv](https://raw.githubusercontent.com/datasets/introduction-to-k-means-Clustering/master/Data/data_1024.csv).
- Na Figura 6 apresentamos um diagrama de dispersão dos dados, segundo essas duas variáveis, mostrando claramente dois grupos distintos: grupo 1, contendo os motoristas que fazem entregas no setor urbano e grupo 2, contendo motoristas que fazem entregas no setor rural.

## K-médias: Exemplo 4

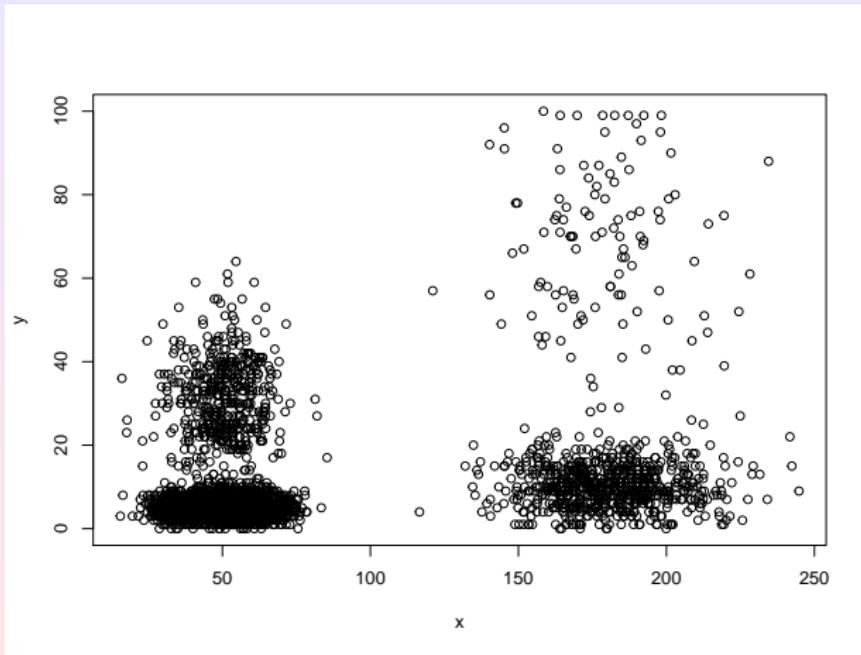


Figura 6: Gráfico de dispersão de  $X_1$  versus  $X_2$  para o Exemplo 4.

## K-médias: Exemplo 4

Utilizando o algoritmo K-médias com  $K = 2$ , obtemos:

Grupo 1: Centróide = (50,05, 8,83)

Grupo 2: Centróide = (180,02, 18,29)

Na Figura 7 temos os dois grupos representados.

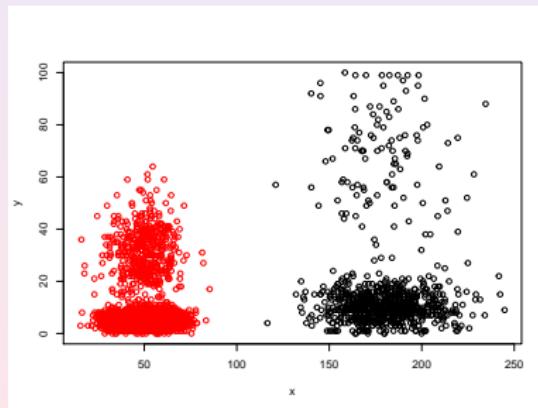


Figura 7: Grupos para o Exemplo 4 com  $K=2$ .

## K-médias: Exemplo 4

Se tomarmos  $K = 4$ , obtemos o gráfico da Figura 8. Agora, os motoristas são separados por aqueles que seguem ou não a velocidade limite, além da divisão zona urbana/rural.

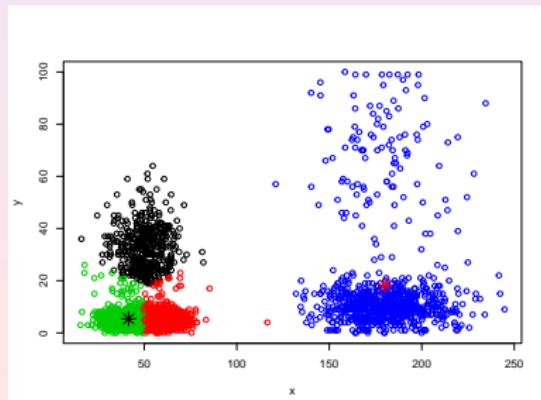
Os centróides são:

Grupo 1: (50,61, 33,06),

Grupo 2: (57,90, 5,28),

Grupo 3: (41,52, 5,40),

Grupo 4: (180,10, 18,31).



## K-médias: Exemplo 5

- **Exemplo 5:** Vamos, agora, considerar dados do Uber, na cidade de Nova Iorque (NYC). Esses dados podem ser obtidos no site

[www.kaggle.com/fivethirtyeight/  
uber-pickups-in-new-york-city/  
data](http://www.kaggle.com/fivethirtyeight/uber-pickups-in-new-york-city/data)

e contém cerca de 4,5 milhões de corridas do Uber de abril a setembro de 2014, além de outros dados do Uber de 2015 e outras de companhias.

- NYC tem 5 distritos: Brooklyn, Queens, Manhattan, Bronx e Staten Island. O conjunto de dados que vamos usar, de 2014, tem informação detalhada sobre a localização do início da corrida com as seguintes colunas:  
Date/Time: dia e hora do início da corrida;  
Lat: a latitude da localidade;  
Lon: a longitude da localidade;  
Base: o código da base da companhia afiliada àquela corrida.
- Os nomes dos arquivos são da forma `uber-raw-data-month.csv`, em que **month** deve ser substituído por `apr14`, `aug14`, `jul14`, `jun14`, `may14`, `sept14`. Em nosso exemplo, vamos usar somente os dados de abril de 2014.

## K-médias: Exemplo 5

- Para ler os dados usamos o comando:

```
> read.csv("https://raw.githubusercontent.com/fivethirtyeight/  
uber-tlc-foil-response/master/uber-trip-data/uber-raw-data-apr14.csv")
```

- Iremos usar o pacote **kmeans** do R e, dependendo do que se quer, outros pacotes, como **dplyr**, **VIM**, **lubridate** ou **ggmap**, poderão ser empregados.
- Com o comando **summary(apr14)**, obtemos:

```
summary(apr14)
      Date.Time           Lat             Lon
4/7/2014 20:21:00 :   97   Min.   :40.07   Min.   :-74.77
4/7/2014 20:22:00 :   87   1st Qu.:40.72   1st Qu.:-74.00
4/30/2014 17:45:00:   78   Median :40.74   Median :-73.98
4/30/2014 18:43:00:   70   Mean    :40.74   Mean   :-73.98
4/30/2014 19:00:00:   70   3rd Qu.:40.76   3rd Qu.:-73.97
4/30/2014 16:55:00:   67   Max.    :42.12   Max.   :-72.07
(Other)                  :564047
```

## K-médias: Exemplo 5

Para ver os dados correspondentes (até dia), temos:

```
head(apr14, n=10)
  Date.Time      Lat      Lon     Base Year Month Day
1 2014-04-01 00:11:00 40.7690 -73.9549 B02512 2014     4    1
2 2014-04-01 00:17:00 40.7267 -74.0345 B02512 2014     4    1
3 2014-04-01 00:21:00 40.7316 -73.9873 B02512 2014     4    1
4 2014-04-01 00:28:00 40.7588 -73.9776 B02512 2014     4    1
5 2014-04-01 00:33:00 40.7594 -73.9722 B02512 2014     4    1
6 2014-04-01 00:33:00 40.7383 -74.0403 B02512 2014     4    1
7 2014-04-01 00:39:00 40.7223 -73.9887 B02512 2014     4    1
8 2014-04-01 00:45:00 40.7620 -73.9790 B02512 2014     4    1
9 2014-04-01 00:55:00 40.7524 -73.9960 B02512 2014     4    1
10 2014-04-01 01:01:00 40.7575 -73.9846 B02512 2014     4    1
```

## K-médias: Exemplo 5

Finalmente, usando o `kmeans` e `ggmap` obtemos a Figura 34. Detalhes dos scripts necessários, estão na página do livro.

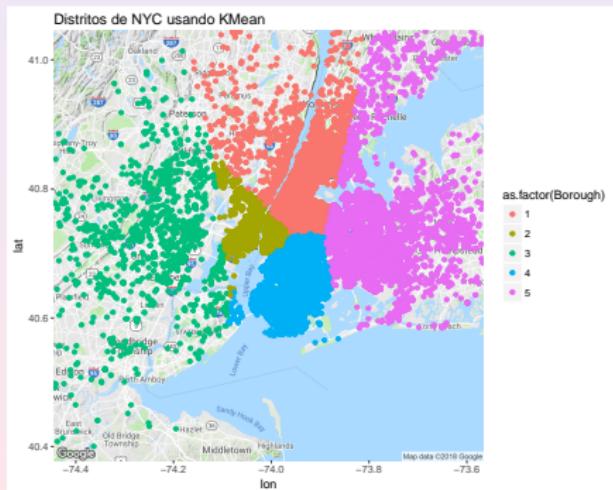


Figura 9: Grupos obtidos para o Exemplo 5.

## Matriz Cofenética

- Esta matriz, **S**, digamos, contém as distâncias entre os objetos a partir do dendograma. Por exemplo, a distância entre os pontos C e L é dada pelo nível em que os dois foram agrupados, nesse caso, 2,24.
- A seguir, verificamos a proximidade entre essa matriz e a matriz de proximidade, **D**.
- Essa proximidade é dada pelo coeficiente de correlação entre os valores de **D** e de **S**, chamado de **coeficiente de correlação cofenético**.
- No Exemplo 12.1, esse valor é 0,80, e pode ser considerado um valor adequado.

## AA–Outros algoritmos

Vimos exemplos com duas variáveis. Podemos ter mais variáveis, mas seu número deve ser menor que o número de indivíduos. Quando temos mais do que duas dimensões, alguns algoritmos foram propostos e são, basicamente, variantes de algoritmos hierárquicos e de  $K$ -means.

- Algoritmo BFR** [(Bradley, Fayyad and Reina, (1998)]. Esse algoritmo é uma variante do  $K$ -means, para o caso de um espaço Euclidiano de alta dimensão mas é baseado numa suposição muito forte: a forma dos agrupamentos segue uma distribuição normal, em cada dimensão, ao redor do respectivo centróide, e além disso as dimensões são independentes. Então, a forma dos grupos deve ser uma elipsóide, com os eixos paralelos aos eixos da dimensão, podendo eventualmente ser um círculo, mas não podem, por exemplo, terem os eixos do elipsóide oblíquos aos eixos da dimensão ou terem formas mais complicadas. Esse algoritmo não está contemplado no R.
- Algoritmo CURE** (de *Clustering Using Representatives*). Esse algoritmo usa procedimentos hierárquicos e os grupos podem ter quaisquer formas mas também não está disponível no R. No entanto, ele pode ser obtido no pacote **pyclustering**, que usa as linguagens Python e C++.

## AA–Outros algoritmos

- c) **Density-based algorithms** (DBSCAN): dado um conjunto de pontos em algum espaço, esse algoritmo agrupa pontos que estão dispostos em uma vizinhança com maior densidade, colocando como *outliers* os pontos que estão em regiões de baixa densidade. Veja Ester et al. (1996).

## AA-distância para strings

- Como salientamos anteriormente, para espaços não Euclidianos (ENE) há outras formas de distância.
- No caso de sequências (**strings**)  $x = x_1x_2 \cdots x_n$  e  $y = y_1y_2 \cdots y_n$ , uma distância conveniente é a distância **edit**, que dá o número de inserções e exclusões de caracteres que converterão  $x$  em  $y$ .
- Por exemplo, considere  $x = abcd$  e  $y = acde$ . A distância **edit** entre essas sequências é  $d_e(x, y) = 2$ , pois temos que excluir  $b$  e inserir  $e$  depois do  $d$ .
- Uma outra maneira de obter essa distância é considerar uma subsequência mais longa comum (SML) a  $x$  e  $y$ . No exemplo, é  $acd$ . Então, a distância **edit** é dada por

$$d_e(x, y) = \ell(x) + \ell(y) - 2\ell(\text{SML}),$$

em que  $\ell$  indica o comprimento de cada sequência. No exemplo,  
 $d_e(x, y) = 4 + 4 - 2 \times 3 = 2$ .

## AA-distância para strings

- Outra distância que pode ser usada é a **distância Hamming**, que dá o número de componentes em que dois vetores (de mesma dimensão) diferem. Por exemplo, se  $x = 110010$  e  $y = 100101$ , então a distância Hamming entre eles é  $d_H(x, y) = 4$ .
- Um problema em ENE é representar um grupo, pois não podemos, por exemplo, calcular o centróide de dois pontos. No exemplo acima, como obter uma sequência entre  $x$  e  $y$ ? Usando a distância *edit* poder-se-ia selecionar algo parecido ao centróide (**o grupóide**), escolhendo-se a sequência que minimiza, por exemplo, a soma das distâncias dessa com as outras sequências do grupo previamente selecionado.

## AA-algoritmos aglomerativos

Considere os algoritmos hierárquicos e dois grupos quaisquer,  $A$  e  $B$ , e a distância entre eles,  $d(A, B)$ . Como vimos, comumente usamos algoritmos aglomerativos que ainda podem ser divididos em (Hastie et al., 2009):

- a) algoritmos com ligação simples (**single linkage**), para os quais toma-se a distância mínima entre os pares, ou seja,

$$d_{SL}(A, B) = \min_{i \in A, j \in B} d(i, j). \quad (6)$$

Essa técnica é também conhecida como técnica do **vizinho mais próximo**;

- b) algoritmos com ligação completa (**complete linkage**), para os quais toma-se a máxima distância entre os pares:

$$d_{CL}(A, B) = \max_{i \in A, j \in B} d(i, j); \quad (7)$$

## AA-algoritmos aglomerativos

- c) algoritmos com ligação média (**group average**), que toma a distância média entre os grupos:

$$d_{GA}(A, B) = \frac{1}{N_A N_B} \sum_{i \in A} \sum_{j \in B} d(i, j). \quad (8)$$

Aqui,  $N_A$  e  $N_B$  indicam os números de observações em cada grupo.

Ligações simples produzem grupos com diâmetros grandes e ligações completas produzem grupos com diâmetros pequenos; agrupamentos por médias representam um compromisso entre esses dois extremos.

## Referências

- Hastie, T., Tibshirani, R. and Friedman, J. (2017). *The Elements of Statistical Learning*, 2nd Edition, Springer.
- James, G., Witten, D., Hastie, T. and Tibshirani, R. (2017). *Introduction to Statistical Learning*. Springer.
- Morettin, P. A. e Singer, J. M. (2022). *Estatística e Ciência de Dados*. LTC: Rio de Janeiro.

# MAE 5905: Introdução à Ciência de Dados

Pedro A. Morettin

Instituto de Matemática e Estatística  
Universidade de São Paulo  
pam@ime.usp.br  
<http://www.ime.usp.br/~pam>

## Aula 13

23 de maio de 2023

# Sumário

1 Redução da dimensionalidade

2 Análise de componentes principais

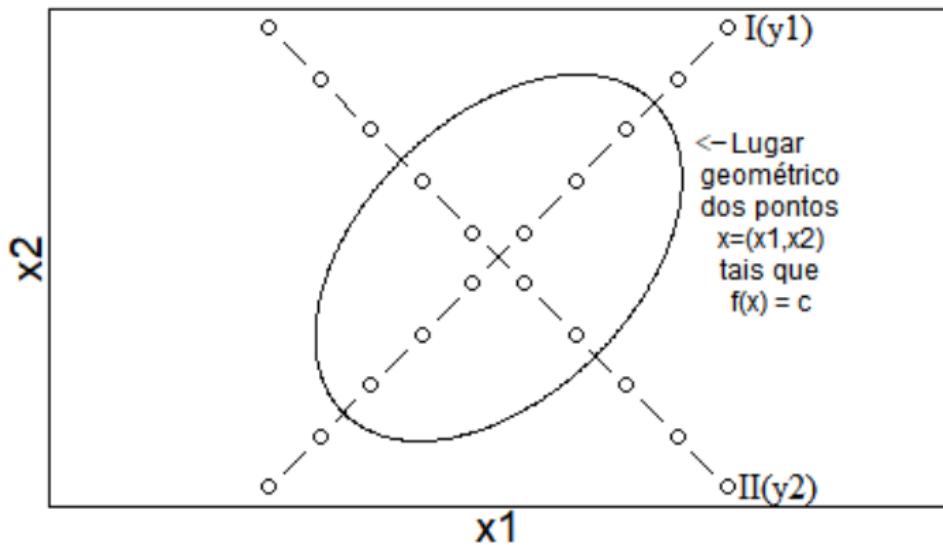
## Preliminares

- As técnicas de Análise de Componentes Principais (ACP), Análise de Fatorial (AF) e Análise de Componentes Independentes (ACI) têm como objetivo reduzir a dimensionalidade de observações multivariadas com base em sua estrutura de dependência.
- A ideia que as permeia é a obtenção de poucos fatores, obtidos como funções das características observadas, que conservem, pelo menos aproximadamente, a estrutura de covariância das variáveis originais.
- Esses poucos fatores vão substituir as variáveis originais em análises subsequentes, servindo, por exemplo, como variáveis explicativas em modelos de regressão. Por esse motivo, a interpretação dessas novas variáveis é muito importante.
- Dessas técnicas, ACP e AF são bastante conhecidas há muito tempo. A ACI (ou ICA, em Inglês) é mais recente, da década de 1990–2000, e não muito contemplada em textos de Estatística.

# ACP – introdução

- A técnica de Componentes Principais consiste numa transformação ortogonal dos eixos de coordenadas de um sistema multivariado.
- A orientação dos novos eixos é determinada por meio da partição sequencial da variância total das observações em porções cada vez menores de modo que, ao primeiro eixo transformado, corresponda o maior componente da partição da variância; ao segundo eixo transformado, a parcela seguinte e assim por diante.
- Se os primeiros eixos forem tais que uma grande parcela da variância seja explicada por eles, poderemos desprezar os demais e trabalhar apenas com os primeiros em análises subsequentes.
- Consideremos duas variáveis  $X_1$  e  $X_2$  com distribuição normal bivariada com média  $\mu$  e matriz de covariâncias  $\Sigma$ .
- O gráfico correspondente aos pontos em que a função densidade de probabilidade é constante é uma elipse; um exemplo está apresentado na Figura 1. Admitimos dados com distribuição normal apenas para finalidade didática. Em geral, essa suposição não é necessária.

## ACP – introdução



## ACP – metodologia

- À medida em que a correlação entre  $X_1$  e  $X_2$  aumenta, o comprimento do eixo maior da elipse também aumenta e o do eixo menor diminui até que a elipse se degenera em um segmento de reta no caso limite em que as variáveis são perfeitamente correlacionadas, i.e., em que o correspondente coeficiente de correlação linear é igual a 1.
- Na Figura 1, o eixo I corresponde ao eixo maior e o eixo II, ao menor. O eixo I pode ser expresso por intermédio de uma combinação linear de  $X_1$  e  $X_2$ , ou seja

$$Y_1 = \beta_1 X_1 + \beta_2 X_2 \quad (1)$$

- No caso extremo, em que  $X_1$  e  $X_2$  são perfeitamente correlacionadas, toda a variabilidade pode ser explicada por meio de  $Y_1$ .
- Quando a correlação entre  $X_1$  e  $X_2$  não é perfeita,  $Y_1$  explica apenas uma parcela de sua variabilidade. A outra parcela é explicada por meio de um segundo eixo, a saber

$$Y_2 = \gamma_1 X_1 + \gamma_2 X_2. \quad (2)$$

## ACP – metodologia

Na Figura 2 apresentamos um gráfico de dispersão correspondente a  $n$  observações  $(X_{1i}, X_{2i})$  do par  $(X_1, X_2)$ .

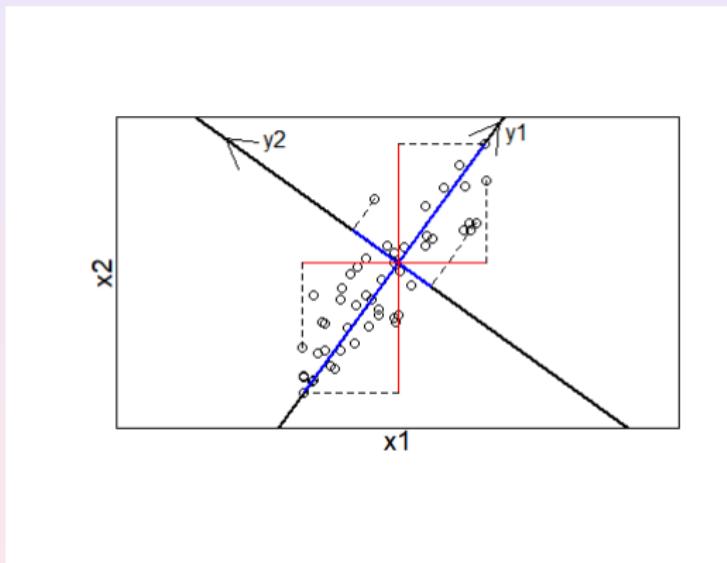


Figura: Gráfico de dispersão de  $n$  observações do par  $(X_1, X_2)$ .

## ACP – metodologia

- A variabilidade no sistema de eixos correspondente a  $(X_1, X_2)$  pode ser expressa como

$$\sum_{i=1}^n (X_{1i} - \bar{X}_1)^2 + \sum_{i=1}^n (X_{2i} - \bar{X}_2)^2$$

em que  $\bar{X}_1$  e  $\bar{X}_2$  são, respectivamente, as médias dos  $n$  valores de  $X_{1i}$  e  $X_{2i}$ .

- No sistema de eixos correspondente a  $(Y_1, Y_2)$ , a variabilidade é expressa como

$$\sum_{i=1}^n (Y_{1i} - \bar{Y}_1)^2 + \sum_{i=1}^n (Y_{2i} - \bar{Y}_2)^2$$

em que  $\bar{Y}_1$  e  $\bar{Y}_2$  têm interpretações similares a  $\bar{X}_1$  e  $\bar{X}_2$ .

## ACP – metodologia

- Se a correlação entre  $X_1$  e  $X_2$  for “grande”, é possível obter valores de  $\beta_1$ ,  $\beta_2$ ,  $\gamma_1$ ,  $\gamma_2$  de tal forma que  $Y_1$  e  $Y_2$  em (1) e (2) sejam tais que

$$\sum_{i=1}^n (Y_{1i} - \bar{Y}_1)^2 \gg \sum_{i=1}^n (Y_{2i} - \bar{Y}_2)^2.$$

- Nesse caso, podemos utilizar apenas  $Y_1$  como variável para explicar a variabilidade de  $X_1$  e  $X_2$ .
- Para descrever o processo de obtenção das componentes principais, consideremos o caso geral em que  $\mathbf{x}_1, \dots, \mathbf{x}_n$  corresponde a uma amostra aleatória de uma variável  $\mathbf{X} = (X_1, \dots, X_p)^\top$  com  $p$  componentes e para a qual o vetor de médias é  $\mu$  e a matriz de covariâncias é  $\Sigma$ .

## ACP – metodologia

- Sejam  $\bar{\mathbf{x}} = n^{-1} \sum_{i=1}^n \mathbf{x}_i$  e  $\mathbf{S} = (n-1)^{-1} \sum_{i=1}^n (\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^\top$ , respectivamente, o vetor de médias amostrais e a matriz de covariâncias amostral.
- A técnica consiste em procurar sequencialmente  $p$  combinações lineares (denominadas **componentes principais**) de  $X_1, \dots, X_p$  tais que à primeira corresponda a maior parcela de sua variabilidade, à segunda, a segunda maior parcela e assim por diante e que, além disso, sejam não correlacionadas entre si.
- A primeira componente principal é a combinação linear

$$Y_1 = \boldsymbol{\beta}_1^\top \mathbf{X} = \beta_{11}X_1 + \dots + \beta_{1p}X_p$$

com  $\boldsymbol{\beta}_1 = (\beta_{11}, \dots, \beta_{1p})^\top$ , para a qual a variância  $Var(Y_1) = \boldsymbol{\beta}_1^\top \boldsymbol{\Sigma} \boldsymbol{\beta}_1$  é máxima.

## ACP – metodologia

- Uma estimativa da primeira componente principal calculada com base na amostra é a combinação linear  $\widehat{Y}_1 = \widehat{\beta}_1^\top \mathbf{x}$  para a qual  $\widehat{\text{Var}}(\widehat{Y}_1) = \widehat{\beta}_1^\top \mathbf{S} \widehat{\beta}_1$  é máxima.
- Este problema não tem solução sem uma restrição adicional, pois se tomarmos  $\widehat{\beta}_1^* = c\widehat{\beta}_1$  com  $c$  denotando uma constante arbitrária, podemos tornar a variância  $\widehat{\text{Var}}(\widehat{Y}_1) = \text{Var}(\widehat{\beta}_1^* \mathbf{x})$  arbitrariamente grande, tomando  $c$  arbitrariamente grande.
- A restrição adicional mais usada consiste em padronizar  $\beta_1$  por meio de  $\beta_1^\top \beta_1 = 1$ .
- Consequentemente, o problema de determinação da primeira componente principal se resume a obter  $\widehat{\beta}_1$  tal que

$$\text{Maximize } \beta_1^\top \Sigma \beta_1, \quad \text{sujeito a } \beta_1^\top \beta_1 = 1.$$

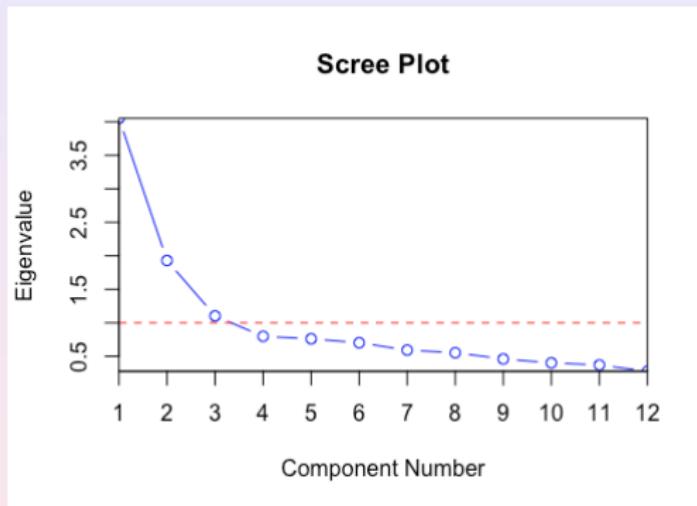
## ACP – metodologia

- A solução desse problema pode ser encontrada por meio da aplicação de **multiplicadores de Lagrange**.
- Dada a primeira componente principal, obtém-se a segunda,  $Y_2 = \beta_2^\top \mathbf{X}$ , por meio da maximização de  $\beta_2^\top \Sigma \beta_2$  sujeito a  $\beta_2^\top \beta_2 = 1$  e  $\beta_1^\top \beta_2 = 0$  (para garantir a ortogonalidade).
- Note que a ortogonalidade das componentes principais implica que a soma de suas variâncias seja igual à variância total do sistema de variáveis. Esse procedimento é repetido até a determinação da  $p$ -ésima componente principal.
- Os coeficientes das componentes principais estimadas são os autovetores  $\hat{\beta}_1, \dots, \hat{\beta}_p$  da matriz  $\mathbf{S}$  e suas variâncias são os autovalores  $\hat{\lambda}_1 \geq \hat{\lambda}_2 \geq \dots \geq \hat{\lambda}_p$  correspondentes.

## ACP – metodologia

- Como a variância total do sistema é  $\text{tr}(\mathbf{S}) = \sum_{i=1}^p \hat{\lambda}_i$ , a contribuição da  $i$ -ésima componente principal é  $\hat{\lambda}_i/\text{tr}(\mathbf{S})$ . Lembrando que  $\mathbf{S} = \sum_{i=1}^p \hat{\lambda}_i \hat{\beta}_i \hat{\beta}_i^\top$  podemos verificar quanto bem ela pode ser aproximada com um menor número de componentes principais. Detalhes podem ser obtidos na Nota de Capítulo 2.
- Na prática, a determinação do número de componentes principais a reter como novo conjunto de variáveis para futuras análises pode ser realizada por meio do **elbow plot**, também conhecido como **scree plot**, que consiste num gráfico cartesiano com os autovalores no eixo vertical e os índices correspondentes às suas magnitudes (em ordem decrescente) no eixo das abscissas.
- Um exemplo está apresentado na Figura 3

# ACP – metodologia



## ACP – metodologia

- A ideia é acrescentar componentes principais até que sua contribuição para a explicação da variância do sistema não apresente contribuições “relevantes”. Com base na Figura 3, apenas as duas primeiras componentes principais poderiam ser suficientes, pois a contribuição das seguintes é apenas marginal.
- Suponhamos que  $r$  componentes principais,  $Y_1, \dots, Y_r$  explicam uma parcela “substancial” da variabilidade do sistema multivariado. Então para a  $k$ -ésima unidade amostral, podemos substituir os valores das variáveis originais  $\mathbf{x}_k = (x_{1k}, \dots, x_{pk})^\top$  pelos correspondentes **escores** associados às componentes principais, nomeadamente,  $\hat{Y}_{1k}, \dots, \hat{Y}_{rk}$  com  $\hat{Y}_{ik} = \hat{\beta}_i^\top \mathbf{x}_k$ .
- Infelizmente, nem a matriz de covariâncias nem os correspondentes autovalores são invariantes relativamente a mudanças de escala. Em outras palavras, mudanças nas unidades de medida das características  $X_1, \dots, X_p$  podem acarretar mudanças na forma e na posição dos elipsóides correspondentes a pontos em que a função densidade é constante.

## ACP – metodologia

- É difícil interpretar combinações lineares de características com unidades de medida diferentes e uma possível solução é padronizá-las por meio de transformações do tipo  $Z_{ij} = (X_{ij} - \bar{X}_i)/S_i$  em que  $\bar{X}_i$  e  $S_i$  representam, respectivamente a média e o desvio padrão de  $X_{ij}$  antes da obtenção das componentes principais.
- A utilização da matriz de correlações **R** obtida por meio dessa transformação pode ser utilizada na determinação das componentes principais; no entanto, os resultados são, em geral, diferentes e não é possível passar de uma solução a outra por meio de uma mudança de escala dos coeficientes.
- Se as características de interesse foram medidas com as mesmas unidades, é preferível extrair as componentes principais utilizando a matriz de covariâncias amostrais **S**.

## ACP – metodologia

- Lembrando que a  $i$ -ésima componente principal é  $Y_i = \beta_{i1}X_1 + \dots + \beta_{ip}X_p$  do sistema multivariado, se as variáveis  $X_1, \dots, X_p$  tiverem variâncias similares ou forem variáveis padronizadas, os coeficientes  $\beta_{ij}$  indicam a importância e a direção da  $j$ -ésima variável relativamente à  $i$ -ésima componente principal.
- Nos casos em que as variâncias das variáveis originais são diferentes, convém avaliar sua importância relativa na definição das componentes principais por meio dos correspondentes coeficientes de correlação. O vetor de covariâncias entre as variáveis originais e a  $i$ -ésima componente principal é

$$\text{Cov}(\mathbf{I}_p \mathbf{X}, \boldsymbol{\beta}_i \mathbf{X}) = \mathbf{I}_p \boldsymbol{\Sigma} \boldsymbol{\beta}_i = \boldsymbol{\Sigma} \boldsymbol{\beta}_i = \lambda_i \boldsymbol{\beta}_i$$

pois  $(\boldsymbol{\Sigma} - \lambda_i \mathbf{I}_p) \boldsymbol{\beta}_i = \mathbf{0}$  (ver Nota de Capítulo 1).

## ACP – metodologia

- Uma estimativa desse vetor de covariâncias é  $\widehat{\lambda}_i \widehat{\beta}_i$ .
- Consequentemente, uma estimativa do coeficiente de correlação entre a  $j$ -ésima variável original e a  $i$ -ésima componente principal é

$$\widehat{Corr}(X_j, \beta_{ij}) = \frac{\widehat{\lambda}_i \widehat{\beta}_{ij}}{\sqrt{\widehat{\lambda}_i} s_j} = \frac{\sqrt{\widehat{\lambda}_i} \widehat{\beta}_{ij}}{s_j}$$

em que  $s_j$  é o desvio padrão amostral de  $X_j$ .

- As componentes principais podem ser encaradas como um conjunto de variáveis latentes (ou fatores) não correlacionados que servem para descrever o sistema multivariado original sem as dificuldades relacionadas com sua estrutura de correlação. Os coeficientes associados a cada componente principal servem para descrevê-las em termos das variáveis originais.
- A comparação entre unidades realizada por meio da componente principal  $Y_i$  é independente da comparação baseada na componente  $Y_j$ . No entanto, essa comparação pode ser ilusória se essas componentes principais não tiverem uma interpretação simples.

## ACP – metodologia

- Como, em geral, isso não é a regra, costuma-se utilizar essa técnica como um passo intermediário para a obtenção de um dos possíveis conjuntos de combinações lineares ortogonais das variáveis originais passíveis de interpretação como variáveis latentes.
- Esses conjuntos estão relacionados entre si por meio de rotações rígidas (transformações ortogonais) e são equivalentes no sentido de aproximar as correlações entre as variáveis originais. Apesar de essas rotações rígidas implicarem uma perda da característica de ordenação das componentes principais em termos de porcentagem de explicação da variabilidade do sistema multivariado, muitas vezes produzem ganhos interpretativos.
- Há muitas opções computacionais para a análise de componentes principais no sistema R, dentre as quais destacamos (funções e pacotes entre parênteses) `prcomp` (stats), `princomp` (stats) e `pca` (FactoMineR). Como há vários métodos tanto para a extração quanto para a rotação das componentes principais, nem sempre os resultados coincidem. A análise deve ser realizada por tentativa e erro tendo como objetivo um sistema com interpretação adequada.

## ACP – Exemplo 1

- **Exemplo 1.** Num estudo em que se pretendia avaliar o efeito de variáveis climáticas na ocorrência de suicídios por enforcamento na cidade de São Paulo foram observadas  $X_1$  = temperatura máxima,  $X_2$  = temperatura mínima,  $X_3$  = temperatura média,  $X_4$  = precipitação e  $X_5$  = nebulosidade diárias para o período de 01/07/2006 e 31/07/2006. Os dados estão disponíveis em <http://www.ime.usp.br/~jmsinger/MorettinSinger/suicidios.xls> e detalhes em Zerbini et al. (2018)].
- Para reduzir o número de variáveis a serem utilizadas como variáveis explicativas numa regressão logística tento como variável resposta a ocorrência de suicídios por enforcamento nesse período consideramos uma análise de componentes principais.
- Os coeficientes das cinco componentes bem como as porcentagem da variância total do sistema explicada por cada uma delas (além da porcentagem acumulada correspondente) estão indicados na Tabela 1.

## ACP – Exemplo 1

**Tabela:** Coeficientes das componentes principais e porcentagens da variância explicada: Exemplo 1

Variável	Componentes principais				
	CP1	CP2	CP3	CP4	CP5
Temperatura máxima	0,93	-0,25	0,05	0,26	0,06
Temperatura mínima	0,91	0,29	-0,18	-0,24	0,06
Temperatura média	0,99	-0,02	-0,03	0,00	-0,11
Precipitação	0,12	0,75	0,66	0,02	0,00
Nebulosidade	-0,12	0,85	-0,50	0,14	-0,01
% Variância	54	28	14	3	0
% Acumulada	54	82	97	100	100

## ACP – Exemplo 1

- Uma análise do gráfico da escarpa sedimentar correspondente representado na Figura 4, conjuntamente com um exame da variância acumulada na Tabela 1 sugere que apenas duas componentes principais podem ser empregadas como resumo, retendo 82% da variância total.
- A primeira componente principal pode ser interpretada como **percepção térmica** e segunda, **percepção de acinzentamento**. Valores dessas novas variáveis para cada unidade amostral são calculadas como  
 $CP_1 = 0,93 * tempmax + 0,91 * tempmin + 0,99 * tempmed$  e  
 $CP_2 = 0,75 * precip + 0,85 * nebul$ , com as variáveis originais devidamente padronizadas.
- O gráfico *biplot* disposto na Figura 5 é conveniente para representar a relação entre as duas componentes principais e as variáveis originais.

## ACP – Exemplo 1

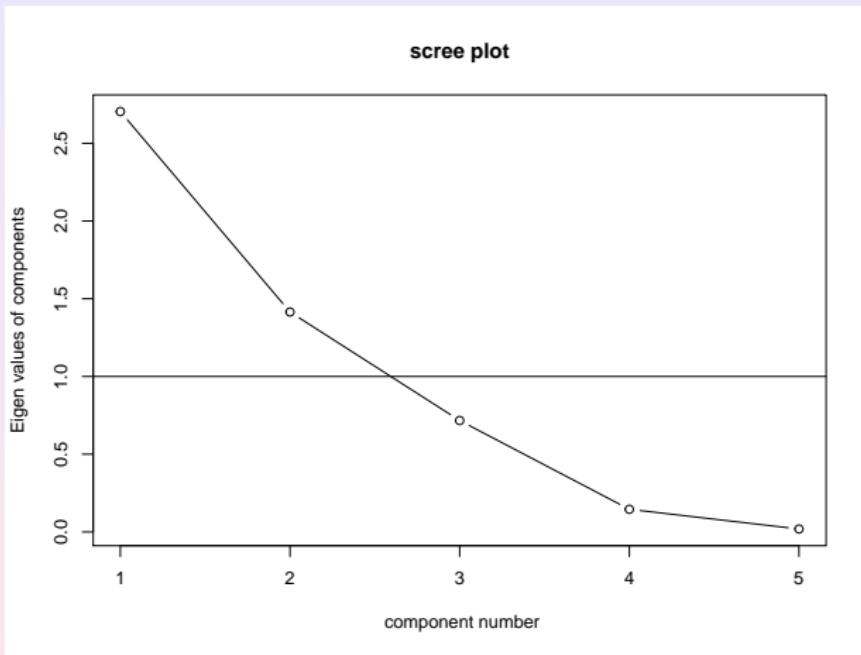


Figura: Gráfico da escarpa sedimentar (ou do cotovelo) para os dados do Exemplo 1.

## ACP – Exemplo 1

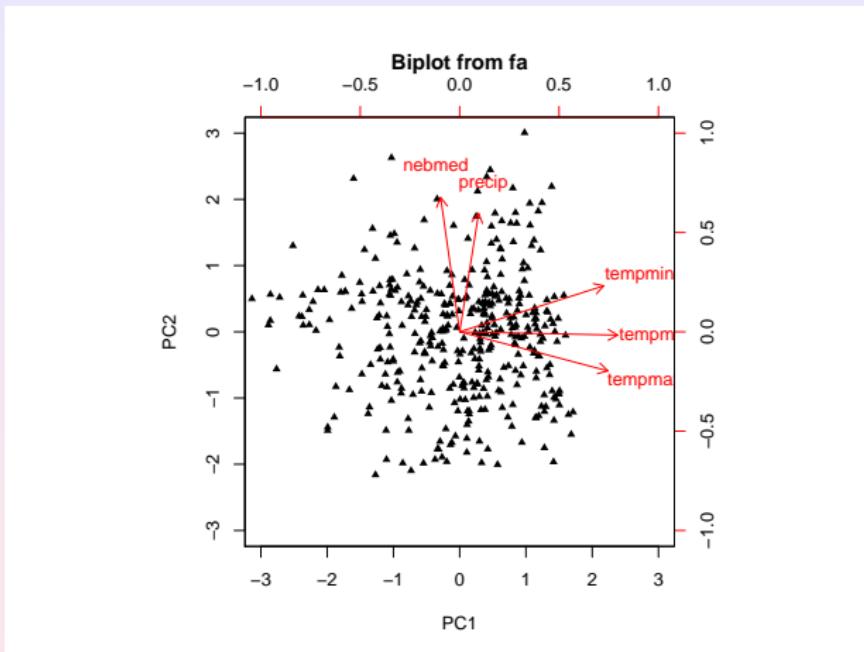


Figura: Gráfico *biplot* para os dados do Exemplo 1.

## Referências

- Hastie, T., Tibshirani, R. and Friedman, J. (2017). *The Elements of Statistical Learning*, 2nd Edition, Springer.
- Härdle, W.K. and Simar, L. (2015). *Applied Multivariate Statistical Analysis*. Springer.
- James, G., Witten, D., Hastie, T. and Tibshirani, R. (2017). *Introduction to Statistical Learning*. Springer.
- Morettin, P. A. e Singer, J. M. (2022). *Estatística e Ciência de Dados*. LTC: Rio de Janeiro.

# MAE 5905: Introdução à Ciência de Dados

Pedro A. Morettin

Instituto de Matemática e Estatística  
Universidade de São Paulo  
pam@ime.usp.br  
<http://www.ime.usp.br/~pam>

## Aula 14

22 de maio de 2023

# Sumário

## 1 Análise Fatorial

## Preliminares

- Primeiramente observemos que para explicar as relações entre  $p$  variáveis são necessárias  $p$  componentes principais; por esse motivo, o modelo adotado não é o ideal.
- O fato de as componentes principais não serem correlacionadas e ordenadas com variâncias decrescentes e o fato de que a técnica corresponde a uma fatoração da matriz de covariâncias da variáveis originais fazem com que a aproximação obtida quando consideramos apenas as primeiras componentes principais seja razoável.
- No entanto, essa técnica pode introduzir um erro sistemático na reprodução das correlações originais, pois podem existir uma ou mais dessas variáveis que sejam muito correlacionadas com as componentes principais desprezadas do que com aquelas retidas na análise.
- Outra observação importante, é que essa técnica utiliza toda a informação sobre cada uma das variáveis originais, embora seja razoável imaginar que uma parcela de sua variabilidade seja específica, nada tendo a ver com as demais variáveis do conjunto sob investigação.
- Além disso, pode-se suspeitar que os “verdadeiros fatores” responsáveis pela geração das observações tenham todos a mesma importância ou que sejam correlacionados entre si.

# Preliminares

- Alguns desses problemas podem ser solucionados por meio da técnica de **Análise Fatorial**. A ideia que a fundamenta está baseada na partição da variância de cada variável do sistema multivariado numa **variância comum** e numa **variância específica**. Além disso, supõe-se que as correlações entre as  $p$  variáveis são geradas por um número  $m < p$  de **variáveis latentes** (ou **fatores**).
- A **vantagem** dessa técnica relativamente àquela de componentes principais está na habilidade de reprodução da estrutura de correlações originais por meio de um pequeno número de fatores sem os erros sistemáticos que podem ocorrer quando simplesmente desprezamos algumas componentes principais.
- As **desvantagens** da Análise Fatorial estão na maior dificuldade de cálculo dos escores fatoriais e na existência de múltiplas soluções. Na realidade a estrutura de correlações das variáveis originais pode ser igualmente reproduzida por qualquer outro conjunto de variáveis latentes de mesma dimensão. A não ser que se imponham restrições adicionais, infinitas soluções equivalentes sempre existirão.

# AF – metodologia

- Consideremos um vetor  $\mathbf{X} = (X_1, \dots, X_p)^\top$  com média  $\boldsymbol{\mu} = (\mu_1, \dots, \mu_p)^\top$  e matriz de covariâncias  $\boldsymbol{\Sigma}$  com elementos  $\sigma_{ij}$ ,  $i, j = 1, \dots, p$ .
- O modelo utilizado para Análise Fatorial de dados provenientes da observação das  $p$  variáveis agrupadas  $X_1, \dots, X_p$  é

$$X_i - \mu_i = \lambda_{i1}F_1 + \dots + \lambda_{im}F_m + e_i = \sum_{j=1}^m \lambda_{ij}F_j + e_i, \quad i = 1, \dots, p \quad (1)$$

em que  $F_j$  é o  $j$ -ésimo **fator comum** a todas as variáveis,  $\lambda_{ij}$  é o parâmetro (chamado de **carga factorial**) que indica a importância desse fator na composição da  $i$ -ésima variável e  $e_j$  é um **fator específico** para essa variável.

- Os coeficientes dos fatores,  $\lambda_{ij}$ , identificam o peso relativo de cada variável no componente. Quanto maior for o valor absoluto do coeficiente, mais importante é a variável correspondente ao estimar o fator. Os **escores fatoriais** são os valores estimados dos fatores.

## AF – metodologia

- Como dito acima, as cargas fatoriais indicam o quanto um fator explica uma variável e variam de -1 a +1.
- Cargas próximas de -1 ou +1 indicam que o fator explica fortemente a variável. Cargas próximas de zero indicam que o fator tem pouca influência sobre a variável.
- Cargas fatoriais são difíceis de interpretar quando não houver rotação. Esta simplifica a estrutura das cargas e torna os fatores mais claramente distinguíveis e fáceis de interpretar.
- Há diversos métodos de rotação (ver abaixo) e devemos escolher aquele que proporciona melhor interpretação.

# AF – metodologia

- Em notação matricial, o modelo pode ser escrito como

$$\mathbf{X} - \boldsymbol{\mu} = \boldsymbol{\Lambda}\mathbf{f} + \mathbf{e} \quad (2)$$

em que  $\boldsymbol{\Lambda}$  é a matriz com dimensão  $p \times m$  de cargas fatoriais,  $\mathbf{f} = (F_1, \dots, F_m)^\top$  é o vetor cujos elementos são os fatores comuns e  $\mathbf{e} = (e_1, \dots, e_p)$  é um vetor cujos elementos são os fatores específicos.

- Adicionalmente, supomos que  $E(\mathbf{f}) = \mathbf{0}$ ,  $Cov(\mathbf{f}) = \mathbf{I}_m$ ,  $E(\mathbf{e}) = \mathbf{0}$  e  $Cov(\mathbf{e}) = \boldsymbol{\psi} = \text{diag}(\psi_1, \dots, \psi_p)$  e que  $Cov(\mathbf{f}, \mathbf{e}) = \mathbf{0}$ . Os elementos de  $\boldsymbol{\psi}$  são as **variâncias específicas**.

## AF – metodologia

- Para avaliar a relação entre a estrutura de covariâncias de  $\mathbf{X}$  e os fatores, observemos que

$$\begin{aligned}
 \text{Cov}(X_i, X_k) &= \text{Cov}\left(\sum_{j=1}^m \lambda_{ij} F_j, \sum_{\ell=1}^m \lambda_{k\ell} F_\ell\right) \\
 &= \sum_{j=1}^m \sum_{\ell=1}^m \lambda_{ij} \lambda_{k\ell} E(F_j F_\ell) + E(e_i e_j) \quad (3) \\
 &= \sum_{j=1}^m \sum_{\ell=1}^m \lambda_{ij} \lambda_{k\ell} + E(e_i e_j)
 \end{aligned}$$

- Consequentemente,  $\text{Cov}(X_i, X_k) = \sigma_{ik} = \sum_{j=1}^m \lambda_{ij} \lambda_{kj}$  se  $i \neq k$  e  $\text{Cov}(X_i, X_i) = \sigma_{ii} = \sum_{j=1}^m \lambda_{ij}^2$ . O termo  $\sum_{j=1}^m \lambda_{ij}^2$  é conhecido por **comunalidade** da  $i$ -ésima variável.
- Em notação matricial, podemos escrever

$$\boldsymbol{\Sigma} = \boldsymbol{\Lambda} \boldsymbol{\Lambda}^\top + \boldsymbol{\psi}$$

e o objetivo é estimar os elementos de  $\boldsymbol{\Lambda}$  e  $\boldsymbol{\psi}$ .

# AF – metodologia

- A comunidade de cada variável é a proporção da variabilidade explicada pelos fatores.
- Quanto mais próxima de 1, melhor é a explicação da variável pelo fator. Decidimos acrescentar um fator se ele contribuir significativamente ao ajuste de algumas variáveis.
- A variância de um fator fornece a variabilidade nos dados explicada pelo fator. Se usarmos CP para extrair fatores, e não usarmos rotação, a variância de cada fator é igual ao seu autovalor.
- Rotação muda a distribuição da proporção da variabilidade explicada por cada fator. Mas a variação total explicada por todos os fatores se mantém.
- Quanto maior a variância de um fator, mais ele explica a variabilidade nos dados. A porcentagem da variância explicada por cada fator varia de zero a 1.

# AF – metodologia

- Em Análise de Componentes Principais, consideramos o modelo linear  $\mathbf{Y} = \mathbf{BX}$  em que  $\mathbf{Y}$  é o vetor cujos elementos são as componentes principais e  $\mathbf{B} = (\beta_1^\top, \dots, \beta_p^\top)^\top$  é a matriz cuja  $i$ -ésima linha contém os coeficientes da  $i$ -ésima componente principal.
- A matriz de covariâncias de  $\mathbf{X}$  é fatorada como  $\Sigma = \Lambda\Lambda^\top$ . Em Análise Fatorial, consideramos o modelo linear  $\mathbf{X} = \Lambda\mathbf{f} + \mathbf{e}$  e a matriz de covariâncias de  $\mathbf{X}$  é fatorada como  $\Sigma = \Lambda\Lambda^\top + \psi$ .
- Uma diferença entre os dois enfoques é que enquanto a fatoração de  $\Sigma$  é única em Análise de Componentes Principais, ela não o é em Análise Fatorial, pois se  $\mathbf{T}$  for uma matriz ortogonal (i.e.,  $\mathbf{TT}^\top = \mathbf{I}_m$ ), obteremos

$$\Sigma = \Lambda\Lambda^\top + \psi = \Lambda\mathbf{T}\mathbf{T}^\top\Lambda^\top + \psi = \Lambda\mathbf{T}(\Lambda\mathbf{T})^\top + \psi$$

e embora as cargas fatoriais  $\Lambda\mathbf{T}$  sejam diferentes das cargas fatoriais  $\Lambda$ , a habilidade de reproduzir a matriz de covariâncias  $\Sigma$  não se altera. Escolhendo matrizes ortogonais diferentes, podemos determinar cargas fatoriais diferentes. A escolha de uma transformação conveniente será discutida posteriormente.

# AF – metodologia

Uma análise factorial consiste dos seguintes passos:

- a) Estimação dos parâmetros do modelo ( $\lambda_{ij}$  e  $\psi_i$ ) a partir de um conjunto de observações das variáveis  $X_1, \dots, X_p$ .
- b) Interpretação dos fatores determinados a partir das cargas fatoriais obtidas em a). Com esse objetivo considera-se a **rotação** dos fatores por meio de transformações ortogonais.
- c) Estimação dos valores dos fatores comuns, chamados **escores fatoriais** para cada unidade amostral a partir dos valores das cargas fatoriais e das variáveis observadas.

# AF – metodologia

- Existem duas classes de métodos para estimação dos parâmetros do modelo fatorial. Na primeira classe consideramos o **método de máxima verossimilhança** e na segunda, métodos heurísticos como o **método do fator principal** ou o **método do centroide**.
- Para o método de máxima verossimilhança, supomos adicionalmente que as variáveis  $X_1, \dots, X_p$  seguem uma distribuição normal (multivariada) e que o número de fatores  $m$  é conhecido. Os estimadores são obtidos por meio da solução do sistema de equações (ver Nota de Capítulo 3)

$$\begin{aligned} \mathbf{S}\psi^{-1}\Lambda &= \Lambda(\mathbf{I}_m + \Lambda^\top\psi^{-1}\Lambda) \\ \text{diag}(\mathbf{S}) &= \text{diag}(\Lambda\Lambda^\top + \psi) \end{aligned} \tag{4}$$

que deve ser resolvido por meio de métodos iterativos. Detalhes podem ser encontrados em Morrison (1972).

- Uma das vantagens desse método é que as mudanças de escala das variáveis originais alteram os estimadores apenas por uma mudança de escala.

## AF – metodologia

- Se uma das variáveis  $X_1, \dots, X_p$  for multiplicada por uma constante, os estimadores das cargas fatoriais correspondentes ficam multiplicados pela mesma constante e o estimador da variância específica associada fica multiplicado pelo quadrado da constante. Dessa forma, podemos fazer os cálculos com as variáveis padronizadas (substituindo a matriz de covariâncias amostral  $\mathbf{S}$  pela correspondente matriz de correlações amostrais  $\mathbf{R}$  e posteriormente escrever os resultados em termos das unidades de medida originais).
- O método do fator principal está intimamente relacionado com a técnica utilizada na análise de componentes principais. Segundo esse método, os fatores são escolhidos obedecendo à ordem decrescente de sua contribuição à communalidade total do sistema multivariado.
- Nesse contexto, o processo tem início com a determinação de um fator  $F_1$  cuja contribuição à communalidade total é a maior possível; em seguida, um segundo fator não correlacionado com  $F_1$  e tal que maximize a communalidade residual é obtido. O processo continua até que a communalidade total tenha sido exaurida.

## AF – metodologia

Na prática, as communalidades e as variâncias específicas devem ser estimadas com base nos dados amostrais. Embora existam vários métodos idealizados para essa finalidade, nenhum se mostra superior aos demais. Dentre os estimadores mais comuns para a communalidade de uma variável  $X_i$ , destacamos:

- i) o quadrado do **coeficiente de correlação múltipla** entre a variável  $X_i$  e as demais;
- ii) o maior valor absoluto dos elementos de  $i$ -ésima linha da matriz de correlações amostrais;
- iii) estimadores obtidos de análises preliminares por meio de processos iterativos.

## AF – metodologia

Outro problema prático é a determinação do número de fatores a incluir na análise. Os critérios mais utilizados para esse fim são:

- i) determinação do número de fatores por meio de algum conhecimento *a priori* sobre a estrutura dos dados;
- ii) número de componentes principais correspondentes a autovalores da matriz **R** maiores que 1;
- iii) explicação de certa proporção (escolhida arbitrariamente) da communalidade ou da variância total.

## AF – metodologia

Um algoritmo comumente utilizado para a obtenção das cargas fatoriais e das variâncias específicas é

- i) Obter as  $p$  componentes principais com base na matriz de correlações amostrais  $\mathbf{R}$ .
- ii) Escolher  $m$  fatores segundo um dos critérios mencionados.
- iii) Substituir os elementos da diagonal principal de  $\mathbf{R}$  por estimadores das communalidades correspondentes por meio de um dos métodos descritos acima, obtendo a chamada **matriz de correlações reduzida**,  $\mathbf{R}^*$ .
- iv) Extrair  $m$  fatores da matriz  $\mathbf{R}^*$ , obtendo novos estimadores das communalidades que vão substituir aqueles obtidos anteriormente na diagonal principal.
- v) Repetir o processo dos itens ii) - iv) até que a diferença entre dois conjuntos sucessivos de estimadores das communalidades seja desprezável.

# AF – metodologia

- O método do centroide foi desenvolvido por Thurstone (1947) para simplificar os cálculos mas não é muito utilizado em virtude das recentes facilidades computacionais; os resultados obtidos por intermédio desse método não diferem muito daqueles obtidos pelo método do fator principal.
- Como a interpretação dos fatores numa análise factorial é uma característica importante em aplicações práticas, pode-se utilizar a técnica de **rotação dos fatores** para obter resultados mais palatáveis.
- Consideremos um exemplo em que cinco variáveis  $A, B, C, D$  e  $E$  são representadas num espaço factorial bidimensional conforme a representação da Figura 1.

## AF – metodologia

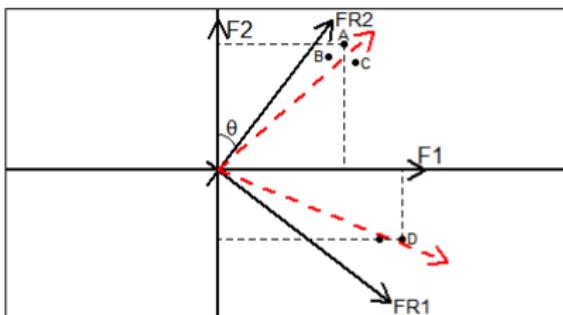


Figura: Representação de cinco variáveis num espaço vetorial bidimensional.

# AF – metodologia

Como ilustrado na Tabela 1, as cargas fatoriais relativas ao fator  $F_1$  são altas e positivas para todas as variáveis. Por outro lado, apenas as variáveis  $A, B$  e  $C$  têm cargas positivas no fator  $F_2$ ; as cargas das variáveis  $D$  e  $E$  são negativas nesse fator.

**Tabela:** Cargas fatoriais para as variáveis  $A, B, C, D$  e  $E$

Variável	Fatores iniciais		Fatores rotacionados	
	$F_1$	$F_2$	$FR_1$	$FR_2$
A	0,75	0,63	0,14	0,95
B	0,69	0,57	0,14	0,90
C	0,80	0,49	0,18	0,92
D	0,85	-0,42	0,94	0,09
E	0,76	-0,42	0,92	0,07

## AF – metodologia

- Dois aglomerados de variáveis podem ser identificados na Figura 1: um formado pelas variáveis  $A, B$  e  $C$  e o outro pelas variáveis  $D$  e  $E$ .
- Apesar disso, esses aglomerados não são evidentes nas cargas fatoriais da Tabela 1. Uma rotação dos fatores (com os eixos rotulados  $FR_1$  e  $FR_2$ ) como aquela indicada na figura juntamente com as novas cargas fatoriais apresentadas na Tabela 1 ressaltam a separação entre os dois conjuntos de variáveis.
- Na solução inicial, cada variável é explicada por dois fatores enquanto que na solução obtida com a rotação dos fatores, apenas um deles é suficiente para explicar a correspondente estrutura de covariância.
- Em princípio, também podemos considerar rotações oblíquas, que são bem mais flexíveis, pois os fatores não precisam ser necessariamente ortogonais. Essa característica pode até ser considerada mais realista, pois a ortogonalidade não é determinante da relação entre os fatores. Os eixos realçados em vermelho na Figura 1 correspondem a uma dessas rotações oblíquas.

## AF – metodologia

O objetivo de qualquer rotação é obter fatores interpretáveis e com a estrutura mais simples possível. Nesse sentido, Thurstone (1947) sugere condições para se obter uma estrutura mais simples, nomeadamente:

- i) Cada linha da matriz de cargas fatoriais  $\Lambda$  deve conter pelo menos um valor nulo.
- ii) Cada coluna da matriz de cargas fatorais deveria ter pelo menos tantos valores nulos quantas forem as colunas.
- iii) Para cada par de colunas deve haver algumas variáveis com cargas fatoriais pequenas numa delas e altas na outra.
- iv) Para cada par de colunas uma grande porcentagem das variáveis deve ter cargas fatoriais não nulas em ambas.
- v) Para cada par de colunas deve haver somente um pequeno número de variáveis com cargas fatoriais altas em ambas.

# AF – metodologia

Como consequência dessas sugestões,

- i) Muitas variáveis (representadas como vetores no espaço dos fatores) devem ficar próximas dos eixos.
- ii) Muitas variáveis devem ficar próximas da origem quando o número de fatores for grande.
- iii) Somente um pequeno número de variáveis ficam longe dos eixos.

## AF – metodologia

A principal crítica às sugestões de Thurstone é que na prática poucas são as situações que admitem uma simplificação tão grande. O que se procura fazer é simplificar as linhas e colunas da matriz de cargas fatorais e os métodos mais comumente empregados com essa finalidade são:

- **Método Varimax** em que se procura simplificar a complexidade fatorial, tentando-se obter fatores com poucos valores grandes e muitos valores nulos ou pequenos na respectiva coluna da matriz de cargas fatorais. Após uma rotação Varimax, cada variável original tende a estar associada com poucos (preferencialmente, um) fatores e cada fator tende a se associar com poucas variáveis. Esse é o método mais utilizado na prática.
- **Método Quartimax** em que se procura maximizar o número de fatores necessários para explicar cada variável. Em geral, esse método produz um fator associado em que muitas variáveis têm cargas altas ou médias, o que nem sempre é conveniente para a interpretação.
- **Método Equimax**, uma mistura dos métodos Varimax e Quartimax.
- **Método Promax**, utilizado para rotações oblíquas.

# AF – estimação

- Um dos objetivos tanto da Análise de Componentes Principais quanto da Análise Fatorial, é substituir as  $p$  variáveis originais  $X_1, \dots, X_p$  por um número menor, digamos,  $m$  em análises subsequentes.
- No caso de componentes principais, podem-se utilizar as estimativas  $\hat{Y}_{ik} = \hat{\beta}_i \mathbf{x}_k$ ,  $i = 1, \dots, m$  para substituir os valores  $\mathbf{x}_k$  observados para a  $k$ -ésima unidade amostral.
- Esse processo é mais complicado quando lidamos com a obtenção dos valores dos fatores  $F_1, \dots, F_m$  (denominados **escores fatoriais**) em Análise Fatorial, que não podem ser estimados no sentido estatístico usual, pois os fatores não são observáveis.
- Com esse objetivo, o **método de Bartlett** (1937) consiste em considerar (2) como um modelo de regressão heterocedátilo em que se supõe que as matrizes de cargas fatoriais,  $\Lambda$  e de variâncias específicas  $\psi$ , são conhecidas e se considera o termo  $e$  como um vetor de erros.

# AF – estimação

Minimizando

$$Q(\mathbf{f}) = \mathbf{e}^\top \boldsymbol{\psi}^{-1} \mathbf{e} = (\mathbf{x} - \boldsymbol{\mu} - \mathbf{\Lambda f})^\top \boldsymbol{\psi}^{-1} (\mathbf{x} - \boldsymbol{\mu} - \mathbf{\Lambda f})$$

obtemos

$$\hat{\mathbf{f}} = [\mathbf{\Lambda}^\top \boldsymbol{\psi}^{-1} \mathbf{\Lambda}]^{-1} \mathbf{\Lambda}^\top \boldsymbol{\psi}^{-1} (\mathbf{x} - \boldsymbol{\mu})$$

e substituindo  $\mathbf{\Lambda}$ ,  $\boldsymbol{\psi}$  e  $\boldsymbol{\mu}$ , respectivamente, por estimativas  $\hat{\mathbf{\Lambda}}$ ,  $\hat{\boldsymbol{\psi}}$  e  $\bar{\mathbf{x}}$ , podemos construir os escores fatoriais para a  $k$ -ésima unidade amostral como

$$\hat{\mathbf{f}}_k = [\hat{\mathbf{\Lambda}}^\top \hat{\boldsymbol{\psi}}^{-1} \hat{\mathbf{\Lambda}}]^{-1} \hat{\mathbf{\Lambda}}^\top \hat{\boldsymbol{\psi}}^{-1} (\mathbf{x}_k - \bar{\mathbf{x}}).$$

# AF – estimação

- Alternativamente, no **método de regressão**, supõe-se que os fatores comuns,  $\mathbf{f}$  e específicos  $\mathbf{e}$  são independentes e têm distribuições normais multivariadas com dimensões  $m$  e  $p$ , respectivamente, de forma que o par  $(\mathbf{X} - \mu, \mathbf{f})$  também tem uma distribuição normal multivariada de dimensão  $p + m$  com matriz de covariâncias

$$\begin{bmatrix} \Lambda \Lambda^\top + \psi & \Lambda \\ \Lambda^\top & I_m \end{bmatrix}.$$

- Utilizando propriedades da distribuição normal multivariada, segue que a distribuição condicional de  $\mathbf{f}$  dado  $\mathbf{X} - \mu$  também é normal multivariada com vetor de médias

$$E(\mathbf{f}|\mathbf{X} - \mu) = \Lambda^\top [\Lambda \Lambda^\top + \psi]^{-1} (\mathbf{X} - \mu)$$

e matriz de covariâncias

$$Cov(\mathbf{f}|\mathbf{X} - \mu) = I_m - \Lambda^\top [\Lambda \Lambda^\top + \psi]^{-1} \Lambda.$$

## AF – Número de fatores

- O termo  $\hat{\Lambda}^T [\hat{\Lambda}\hat{\Lambda}^T + \psi]^{-1}$  corresponde aos coeficientes de uma regressão multivariada tendo os fatores como variáveis respostas e  $\mathbf{X} - \mu$  como variáveis explicativas. Utilizando as estimativas  $\hat{\Lambda}$ ,  $\hat{\psi}$ , podemos calcular os escores fatoriais para a  $k$ -ésima unidade amostral (com valores das variáveis originais  $\mathbf{x}_k$ ) por meio de

$$\hat{\mathbf{f}}_k = \hat{\Lambda}^T [\hat{\Lambda}\hat{\Lambda}^T + \hat{\psi}]^{-1} (\mathbf{x}_k - \bar{\mathbf{x}}).$$

Pacotes do R que podem ser utilizados para a AF são o [psych](#), o [GPArotation](#) e o [robustfa](#).

- Para determinar o número de fatores a considerar, podemos usar vários critérios:
  - [1] **Teste Scree**: devido a Cattel (1966), consiste do gráfico dos autovalores contra o número de fatores (ou CPs, de uma ACP). Procura-se o ponto ([elbow](#)) onde a inclinação muda drasticamente.
  - [2] **Regra de Kaiser–Guttman**: devida a Guttman(1954) e Kaiser (1960), é similar ao teste scree, mas consideram-se componentes ou fatores com autovalores maiores do que 1. Esse é o gráfico que temos usado.
  - [3] **Análise paralela**: devida a Horn (1965), propõe reter somente autovalores que sejam superiores ou iguais à média dos autovalores obtidos de  $k$  matrizes de correlações calculadas com  $n$  observações aleatórias.



# AF – Número de fatores

A estratégia da Análise Paralela é:

- (a) Gere  $n$  v.a.s de acordo com uma  $N(0, 1)$  independentemente para  $p$  variáveis;
- (b) calcule a matriz de correlações de Pearson;
- (c) calcule os autovalores dessa matriz;
- (d) repita passos (a)-(c)  $k$  vezes;
- (e) calcule uma medida de localização, ML, para os  $k$  vetores de autovalores: média, mediana,  $p$ -quantil etc;
- (f) Substitua o valor 1 da regra de Guttman–Kaiser, ou seja, conte o número de autovalores maiores que ML.

Há, também, soluções não gráficas a esses testes.

- Ótima coordenada:

$$n_{oc} = \#\{(\lambda_i \geq 1) \text{ e } (\lambda_i \geq \lambda \text{ previsto pelo teste scree K-G})\}$$

ou

$$n_{oc} = \#\{(\lambda_i \geq ML \text{ e } (\lambda_i \geq \lambda \text{ previsto pelo teste scree da AP})\}$$

- **Fator de aceleração:** coloca ênfase na coordenada onde a inclinação da curva muda abruptamente

## AF – Exemplo 1

- **Exemplo 1.** Num estudo planejado para avaliar o nível de poluição atmosférica por meio de medidas de elementos químicos depositados em cascas de árvores, obtiveram-se observações da concentração de Al, Ba, Cu, Fe, Zn, P, Cl, Sr e Ca entre outros elementos em 193 unidades da espécie *Tipuana tipu* na cidade de São Paulo.
- Esses dados constituem um subconjunto daqueles disponíveis em <http://www.ime.usp.br/~jmsinger/MorettinSinger/arvores.xls> O objetivo aqui é obter um conjunto de fatores que permitam identificar características comuns a essas variáveis. Os resultados provenientes de uma análise de componentes principais estão dispostos na Tabela 2.

## AF – Exemplo 1

Tabela: Coeficientes de componentes principais (CP) para os dados do Exemplo 1

	CP1	CP2	CP3	CP4	CP5	CP6	CP7	CP8	CP9
Al	0.90	-0.11	0.09	0.21	-0.16	0.19	-0.08	-0.17	0.17
Ba	0.88	-0.16	0.09	0.10	-0.10	0.27	0.09	0.31	-0.01
Cu	0.82	0.18	-0.05	-0.23	0.31	-0.18	-0.31	0.08	0.01
Fe	0.95	-0.10	0.07	0.10	-0.03	0.10	0.00	-0.18	-0.19
Zn	0.83	0.16	-0.13	-0.22	0.29	-0.21	0.31	-0.05	0.05
P	0.25	0.69	-0.25	0.53	-0.20	-0.28	0.00	0.05	-0.01
Cl	0.17	0.53	0.60	-0.42	-0.39	-0.07	0.01	0.00	0.00
Mg	-0.24	0.22	0.78	0.35	0.40	0.05	0.02	0.00	0.00
Ca	-0.20	0.77	-0.33	-0.12	0.15	0.47	0.00	-0.04	0.00
% Var	0.45	0.17	0.13	0.08	0.07	0.06	0.02	0.02	0.01
% Acum	0.45	0.61	0.75	0.83	0.89	0.95	0.97	0.99	1.00

## AF – Exemplo 1

Uma análise da porcentagem da variância explicada pelas componentes principais juntamente com um exame do gráfico da escarpa sedimentar (**scree plot**) correspondente, apresentado na Figura 2 sugere que três fatores, que explicam 75% da variância total do sistema de variáveis originais poderiam contemplar uma representação adequada.

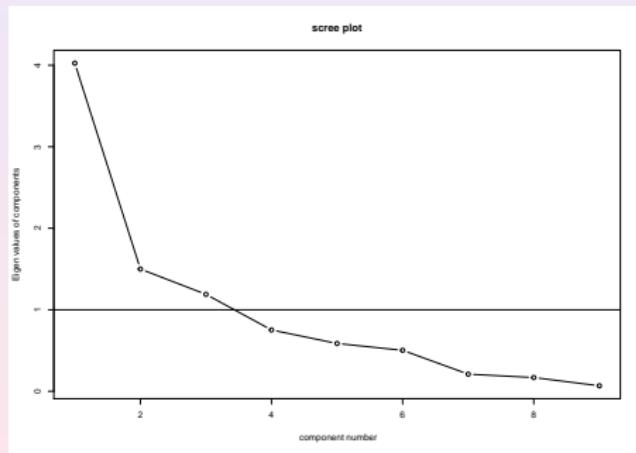


Figura: Gráfico da escarpa sedimentar para os dados do Exemplo 1.

## AF – Exemplo 1

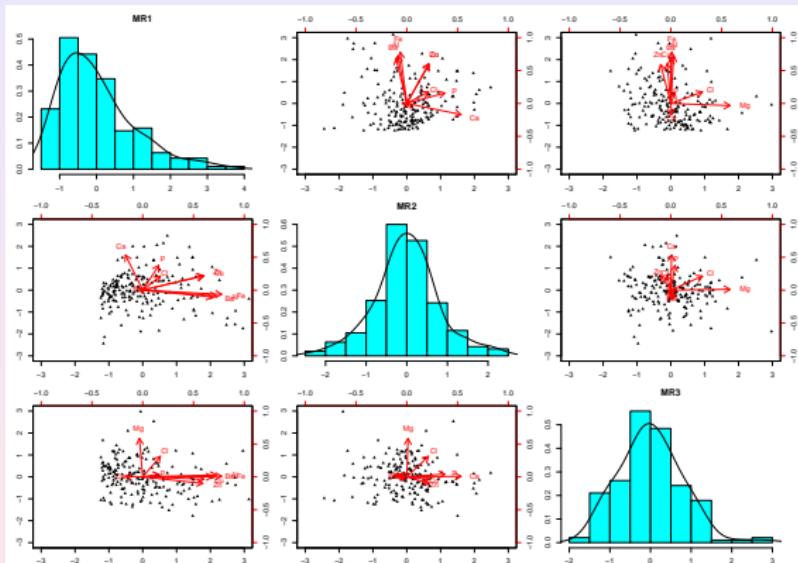
As cargas fatoriais correspondentes três fatores rotacionados obliquamente juntamente com as communalidades e especificidades correspondentes estão dispostos na Tabela 3.

**Tabela:** Cargas fatoriais, communalidades e especificidades correspondentes a uma análise factorial para os dados do Exemplo 1

	Fator 1	Fator 2	Fator 3	Comunalidade	Especificidade
Al	0.91	-0.11	0.03	0.82	0.18
Ba	0.86	-0.13	0.00	0.75	0.25
Cu	0.75	0.27	-0.04	0.66	0.34
Fe	0.97	-0.08	0.01	0.95	0.05
Zn	0.74	0.27	-0.13	0.68	0.32
P	0.20	0.47	0.05	0.25	0.75
Cl	0.22	0.27	0.39	0.22	0.78
Mg	-0.04	0.02	0.73	0.54	0.46
Ca	-0.21	0.67	0.01	0.49	0.51

# AF – Exemplo 1

Os gráficos das Figuras 3 e 4 também podem ser utilizados para identificar os fatores.



**Figura:** Gráfico *biplot* correspondente aos três fatores considerados para os dados do Exemplo 1.

## AF – Exemplo 1

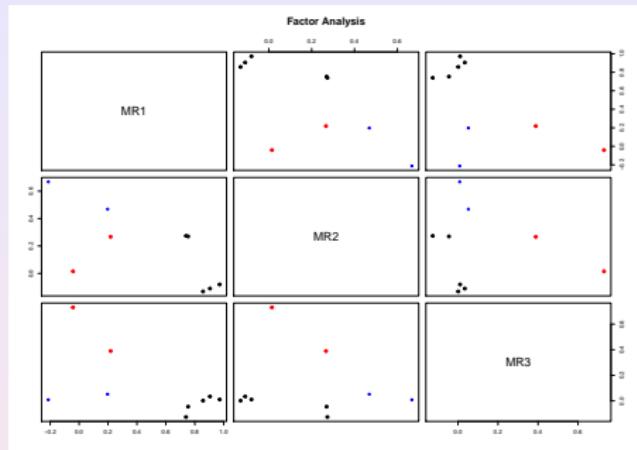


Figura: Gráfico cartesiano correspondente aos três fatores considerados para os dados do Exemplo 1.

O Fator 1 tem cargas fatoriais concentradas nos elementos Al, Ba, Cu, Fe e Zn, que estão associados à poluição de origem veicular gerada por desgaste de freios e ressuspensão de poeira; o Fator 2 está tem cargas fatoriais concentradas em P e Ca, características da saúde arbórea e o Fator 3 tem as cargas fatoriais concentradas em Cl e Mg.

## AF – Exemplo 2

- **Exemplo 2.** Vamos retomar o Exemplo 1 da Aula 13, em que se pretendia avaliar os efeitos de variáveis climáticas na ocorrência de suicídios por enforcamento na cidade de São Paulo.
- Nesse exemplo obtivemos duas CPs que explicavam 82% da variância total.
- Vamos usar a função `factanal()` do pacote `stats`. Essa função usa MV numa matriz de covariâncias ou numa matriz de dados.
- Obtemos os resultados abaixo:

Call:

```
factanal(x = clim1, factors = 2, rotation = "varimax")
```

Uniquenesses:

	tempmax	tempmin	tempmed	precip	nebmed
	0.061	0.020	0.005	0.890	0.482

Loadings:

	Factor1	Factor2
tempmax	0.932	-0.266
tempmin	0.883	0.447
tempmed	0.997	
precip		0.323
nebmed	-0.142	0.706

	Factor1	Factor2
SS loadings	2.668	0.874
Proportion Var	0.534	0.175
Cumulative Var	0.534	0.708

Test of the hypothesis that 2 factors are sufficient.  
The chi square statistic is 11.03 on 1 degree of freedom.  
The p-value is 0.000895

## AF – Exemplo 2

- Na saída, encontramos o termo **uniqueness**, ou **ruído**, corresponde à proporção da variabilidade que **não pode ser explicada** pela combinação linear dos fatores(variância específica). Valor alto para uma variável indica que o fator não contribui bem para a sua variância. É a diagonal da matriz  $\psi$ . Nesse caso, os fatores não contribuem para a variância de Precipitação e Nebulosidade.

- A matriz  $\lambda$  das cargas fatoriais é dada a seguir:

```
> load<-fafit\$loadings[,1:2]
      Factor1    Factor2
tempmax  0.93205114 -0.26553163
tempmin  0.88305960  0.44690513
tempmed  0.99713621  0.02715943
precip   0.07422757  0.32313465
nebmed   -0.14166420  0.70579832
```

- A matriz  $\psi$  é dada por

```
> Psi <-diag(fafit\$uniquenesses)
```

	[,1]	[,2]	[,3]	[,4]	[,5]
[1,]	0.0607741	0.0000000	0.000	0.0000	0.0000000
[2,]	0.0000000	0.02048162	0.000	0.0000	0.0000000
[3,]	0.0000000	0.0000000	0.005	0.0000	0.0000000
[4,]	0.0000000	0.0000000	0.000	0.8901	0.0000000
[5,]	0.0000000	0.0000000	0.000	0.0000	0.4817635

- A comunalidade é obtida tomando os quadrados das cargas:

```
> apply(fafit\$loadings^2,1,sum)
tempmax tempmin tempmed precip nebmed
```

## AF – Exemplo 2

- Não foi possível considerar 3 fatores, pois o programa não aceita valor maior do que 2 para 5 variáveis.
- A proporção da variância explicada pelos fatores é 70,8%, menor do que as CPs.
- A estimativa da matriz  $\Sigma$  é dada por

	tempmax	tempmin	tempmed	precip
tempmax	1.00000048	0.7043893	0.92217026	-0.01661858
tempmin	0.70438926	1.0000001	0.89266839	0.20995790
tempmed	0.92217026	0.8926684	1.00001826	0.08279115
precip	-0.01661858	0.2099579	0.08279115	1.00002576
nebmed	-0.31945006	0.1903270	-0.12208942	0.21755251
	nebmed			
tempmax	-0.3194501			
tempmin	0.1903270			
tempmed	-0.1220894			
precip	0.2175525			
nebmed	0.9999835			

## AF – Exemplo 2

- A matriz residual estimada é

	tempmax	tempmin	tempmed	precip	nebmed
tempmax	0.000000	-0.000390	0.000104	-0.015332	0.009215
tempmin	-0.000390	0.000000	0.000035	-0.008720	0.000870
tempmed	0.000104	0.000035	-0.000018	0.003171	-0.000927
precip	-0.015332	-0.008720	0.003171	-0.000026	0.072404
nebmed	0.009215	0.000870	-0.000927	0.072404	0.000017

e vemos que os valores são próximos de zero, indicando que o modelo fatorial está adequado.

- Para determinar o número de fatores podemos obter os autovalores da matriz de correlações estimada:

```
> ev<-eigen(cor(clima1))  
eigen() decomposition  
values
```

```
[1] 2.70399548 1.41461150 0.71677349 0.14576231 0.01885722
```

mostrando que tomamos 2 fatores, correspondentes aos valores próprios maiores que 1.

## AF – Exemplo 2

A seguir temos alguns gráficos mencionados anteriormente sobre a determinação do número de fatores.

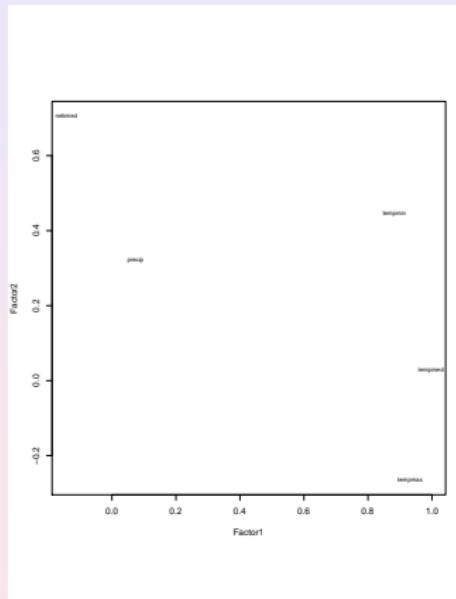


Figura: Gráfico do fator 1 vs fator 2 para o Exemplo2

## AF – Exemplo 2

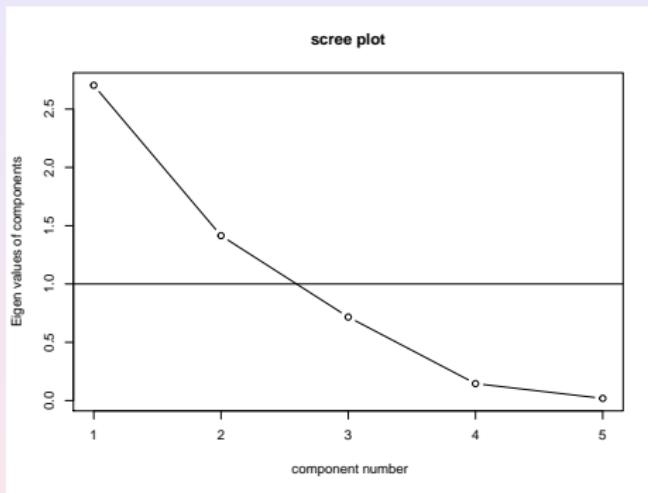


Figura: Scree plot para Exemplo 2.

## AF – Exemplo 2

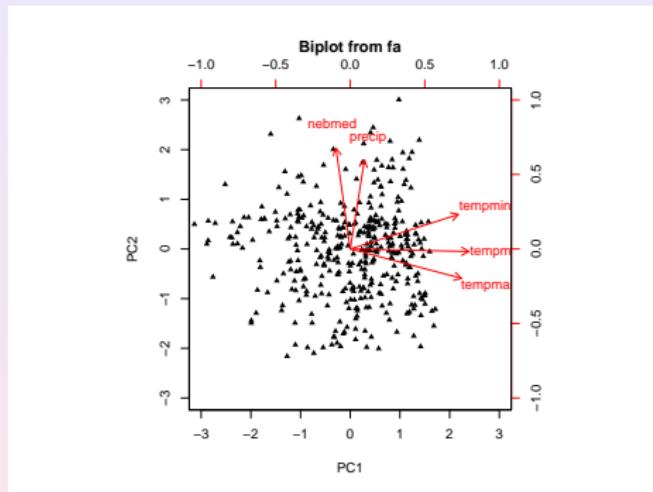


Figura: Biplot para Exemplo 2.

## AF – Exemplo 2

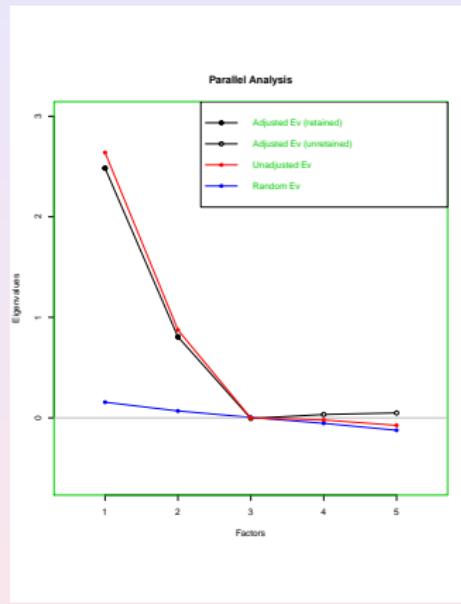


Figura: Análise Paralela para Exemplo 2.

## AF – Exemplo 2

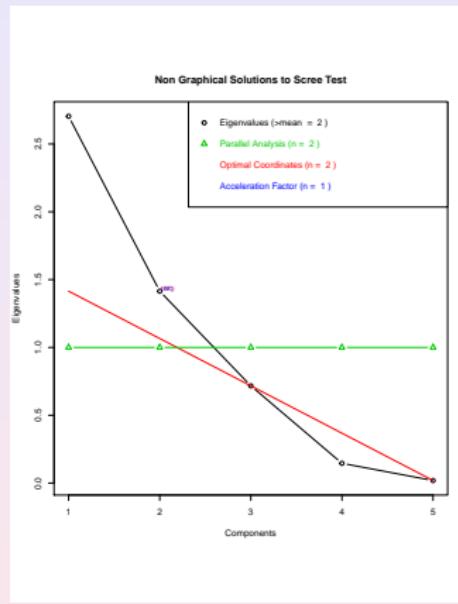


Figura: Soluções não gráficas para Exemplo 2.

## Referências

- Bartlett, M.S. (1937). The statistical conception of mental factors. *British Journal of Psychology*, **28**, 97-104.
- Hastie, T., Tibshirani, R. and Friedman, J. (2017). *The Elements of Statistical Learning*, 2nd Edition, Springer.
- Härdle, W.K. and Simar, L. (2015). *Applied Multivariate Statistical Analysis*. Springer.
- James, G., Witten, D., Hastie, T. and Tibshirani, R. (2017). *Introduction to Statistical Learning*. Springer.
- Morettin, P. A. e Singer, J. M. (2022). *Estatística e Ciência de Dados*. LTC: Rio de Janeiro.
- Morrison, D.F. (1976). *Multivariate Statistical Methods*, 2nd Ed. New York: McGraw-Hill.
- Thurstone, L.L. (1947). *Multiple Factor Analysis: A development and expansion of vectors of the mind..* Chicago: University of Chicago Press.

# MAE 5905: Introdução à Ciência de Dados

Pedro A. Morettin

Instituto de Matemática e Estatística  
Universidade de São Paulo  
[pam@ime.usp.br](mailto:pam@ime.usp.br)  
<http://www.ime.usp.br/~pam>

## Aula 15

7 de junho de 2023

# Sumário

1 Análise de Componentes Independentes

2 Decomposição de matrizes

## Preliminares

- Vimos, no estudo de componentes principais, que essas são não correlacionadas, ou ortogonais. No caso da análise fatorial, há a suposição adicional de normalidade. Na análise de componentes independentes (ICA, em Inglês), a ideia é obter componentes independentes e não gaussianas.
- A ICA é relativamente recente, sendo introduzida na década de 1980 no contexto de redes neurais e encontra aplicações em processamento de sinais biomédicos, separação de sinais de áudio, séries temporais financeiras etc.
- As referências principais nessa área são Hyvärinen e Oja (1997), Hyvärinen (1999), Hyvärinen et al. (1999), Stone (2004) e Gegembauer (2010).
- Vamos supor que os dados consistem de um número  $p$  de variáveis,  $\mathbf{X} = (X_1, \dots, X_p)^\top$ , e consideramos a transformação

$$Y_i = \sum_{j=1}^p w_{ij} X_j(t), \text{ para } i = 1, \dots, q, \quad (1)$$

onde  $q < p$  e os  $w_{ij}$  são alguns coeficientes que definem a representação.

- Reunindo os coeficientes  $w_{ij}$  em uma matrix  $\mathbf{W}$ , de ordem  $q \times p$ , e os  $Y_j$  num vetor  $\mathbf{Y}$ , de ordem  $q \times 1$ , obtemos

$$\mathbf{Y} = \mathbf{WX}$$

## Separação cega de fontes

- A diferença com CP e AF é que escolhemos  $\mathbf{W}$  de modo que as variáveis  $Y_i$  sejam independentes e não gaussianas. ICA é um caso particular da **Separação Cega de Fontes** (*blind source separation*).
- Considere a situação onde existem pessoas conversando na mesma sala, assim emitindo sinais de fala; ou telefones celulares emitindo suas ondas de rádio. Suponha que haja vários sensores ou receptores que estão em diferentes posições, sendo assim cada mistura das fontes originais é gravada com pesos um pouco diferentes.
- Com o objetivo de simplificar a exposição, digamos que existem três fontes que dão origem aos sinais, e também três sinais observados, denotados por  $X_1(t)$ ,  $X_2(t)$  e  $X_3(t)$ , os quais são as amplitudes dos sinais gravados no tempo  $t$ , e por  $S_1(t)$ ,  $S_2(t)$  e  $S_3(t)$  os sinais originais. Os  $X_i(t)$  são uma combinação linear dos  $S_i(t)$  com coeficientes constantes  $a_{ij}$ , os quais dão as misturas dos pesos e que dependem da distância entre as fontes e os sensores e se supõe que sejam desconhecidos:

$$X_1(t) = a_{11}S_1(t) + a_{12}S_2(t) + a_{13}S_3(t), \quad (3)$$

$$X_2(t) = a_{21}S_1(t) + a_{22}S_2(t) + a_{23}S_3(t),$$

$$X_3(t) = a_{31}S_1(t) + a_{32}S_2(t) + a_{33}S_3(t).$$

## Separação cega de fontes

- Em geral, não conhecemos os valores  $a_{ij}$  e a fonte do sinal  $S_i(t)$  também é desconhecida. O que gostaríamos de fazer é encontrar os sinais originais a partir das misturas  $X_i(t)$ . Isso se chama **Separação Cega de Fontes** (*Blind Source Separation*).
- O termo **cega** significa que temos pouca ou nenhuma informação a respeito das fontes.
- Supomos que os coeficientes de mistura  $a_{ij}$  sejam diferentes o suficiente, para tornar a matriz que formam invertível. Então, existe uma matriz **W** com coeficiente  $w_{ij}$ , tal que podemos escrever os  $S_i$  como:

$$S_i(t) = w_{i1}X_1(t) + w_{i2}X_2(t) + w_{i3}X_3(t), \quad i = 1, 2, 3, \quad (4)$$

- Tal matriz **W** pode ser encontrada como a inversa da matriz que consiste dos coeficientes de mistura  $a_{ij}$  em (3) se conhecermos os coeficientes  $a_{ij}$ .

## Separação cega de fontes

- A questão agora é: como podemos estimar os coeficientes  $w_{ij}$  em (4) ? Observamos os sinais  $X_1$ ,  $X_2$  e  $X_3$  e queremos achar uma matriz  $\mathbf{W}$  de modo que a representação seja dada pelos sinais da fonte original  $S_1$ ,  $S_2$  e  $S_3$ .
- Uma solução simples ao problema pode ser encontrada, considerando independência estatística dos sinais. De fato, se os sinais são não gaussianos, isto é suficiente para determinar os coeficientes  $w_{ij}$  dado que os sinais

$$Y_i(t) = w_{i1}X_1(t) + w_{i2}X_2(t) + w_{i3}X_3(t), \quad i = 1, 2, 3, \quad (5)$$

sejam estatisticamente independentes. Se os sinais  $Y_i$ , são independentes, então eles são iguais aos sinais originais  $S_i$ ,  $i = 1, 2, 3$  (a menos da multiplicação por um escalar). Usando apenas esta informação de independência estatística, podemos de fato estimar os coeficientes da matriz  $\mathbf{W}$ .

## ICA – metodologia

- O que vimos acima nos leva a seguinte definição da ICA. Dado um vetor de variáveis aleatórias  $\mathbf{X} = (X_1, X_2, \dots, X_p)^\top$ , supõe-se que estas sejam geradas como uma mistura linear de componentes independentes  $\mathbf{S} = (S_1, S_2, \dots, S_p)^\top$ :

$$\begin{pmatrix} X_1 \\ X_2 \\ \vdots \\ X_p \end{pmatrix} = \mathbf{A} \begin{pmatrix} S_1 \\ S_2 \\ \vdots \\ S_p \end{pmatrix}, \quad (6)$$

ou

$$\mathbf{X} = \mathbf{AS}, \quad (7)$$

onde  $\mathbf{A} = [a_{ij}]$  é uma matriz desconhecida (chamada *mixing*).

- A análise de componentes independentes consiste agora em estimarmos tanto a matriz  $\mathbf{A}$  como os  $S_i$ , onde apenas observamos os  $X_i$ . Observe que supomos que o número de componentes independentes  $S_i$  é igual ao número de variáveis observadas; isso é uma simplificação que não é completamente necessária.

## ICA – metodologia

- Pode ser demonstrado que este problema é bem definido, isto é, o modelo em (6) pode ser estimado se e somente se os componentes  $S_i$  são não gaussianos.
- Essa é uma necessidade fundamental que também explica a principal diferença entre ICA e análise fatorial, na qual a normalidade dos dados é levada em conta.
- De fato, ICA pode ser considerada como uma análise fatorial não gaussiana, visto que nesta também modelamos os dados como uma mistura linear de alguns fatores latentes.
- Além da estimação, tem-se que encontrar um algoritmo para executar os cálculos necessários. Como o princípio de estimação usa funções não quadráticas, os cálculos necessários, usualmente, não podem ser expressos usando álgebra linear simples, podendo ser extremamente complexos. Algoritmos numéricos são assim uma parte integral dos métodos de estimação ICA.

## ICA – metodologia

- Os métodos numéricos são tipicamente baseados na otimização de algumas funções objetivas. O método básico de otimização é o método do gradiente. De interesse particular tem-se o algoritmo de ponto fixo chamado **FastICA**, que tem sido usado para explorar uma particular estrutura dos problemas ICA. Por exemplo, podemos usar esses métodos para encontrar o máximo de não normalidade como medido pelo valor absoluto do excesso de curtose.
- Estimando-se a matriz  $\mathbf{A}$ , supondo-se que exista sua inversa  $\mathbf{A}^{-1} = \mathbf{W}$ , obtemos

$$\mathbf{S} = \mathbf{W}\mathbf{X}. \quad (8)$$

- Há duas ambiguidades no procedimento da ICA. A primeira é que não podemos determinar as variâncias das  $S_i$ , pois se as multiplicarmos por uma constante  $C$ , basta dividir as colunas correspondentes de  $\mathbf{A}$  por  $C$ . Dessa maneira, o que fazemos é centrar as componentes, isto é,  $E(S_i) = 0$ , para todo  $i$ , e fixar as magnitudes das componentes independentes (c.i.'s), de modo que  $E(S_i^2) = 1$ , para todo  $i$ .

## ICA – metodologia

- A segunda ambiguidade é que, contrariamente ao que ocorre com a ACP, não podemos determinar a ordem das c.i., pois como  $\mathbf{S}$  e  $\mathbf{A}$  são desconhecidas, podemos mudar livremente a ordem dos termos em (7).
- Por que variáveis gaussianas não são permitidas? Para responder, considere o caso particular (3), suponha que  $\mathbf{A}$  seja **ortogonal** e que as  $S_i(t)$  sejam gaussianas. Então, as variáveis  $X_i(t)$  são gaussianas (combinações lineares de v.a.s independentes e gaussianas), não correlacionadas (por que?) e de variância unitária. Sua densidade conjunta é dada por

$$f(x_1, x_2, x_3) = \frac{1}{(2\pi)^{3/2}} e^{-(x_1^2 + x_2^2 + x_3^2)/2},$$

que é simétrica, logo não contém nenhuma informação sobre as direções das colunas de  $\mathbf{A}$ . Logo,  $\mathbf{A}$  não pode ser estimada, ou ainda,  $\mathbf{A}$  não é identificada para componentes gaussianas independentes.

## ICA – metodologia

- Seja  $y = \mathbf{v}' \mathbf{X} = \sum_{i=1}^p v_i X_i$ , onde  $\mathbf{v}$  é um vetor  $p \times 1$ . Se  $\mathbf{v}$  fosse uma das linhas de  $\mathbf{A}^{-1} = \mathbf{W}$ , essa combinação linear seria uma das c.i.'s  $S_i$ . O problema reduz-se, então, a determinar um vetor  $\mathbf{v}$  nessas condições.
- Pode-se provar que maximizando-se a não gaussianidade de  $\mathbf{v}' \mathbf{X}$  nos dá uma das componentes de  $\mathbf{S}$ . Para tanto, teremos que obter  $2p$  máximos locais, 2 para cada c.i. (correspondendo a  $S_i$  e  $-S_i$ ). Como as c.i.'s são não correlacionadas, podemos restringir a busca no espaço que fornece estimativas não correlacionadas com as anteriores e isso corresponde a **branquear (whitening)** as observações.
- Como obter componentes independentes? Podemos escolher duas formas como proxy de independência , que por sua vez determinam a forma do algoritmo ICA a usar:
  1. Minimização da informação mútua (MIM);
  2. Maximização da não gaussianidade (MNG).
- A família de algoritmos que usa MIM utiliza medidas como a **Divergência de Kullback-Leibler** e **Máxima Entropia**. A família que aborda a não gaussianidade, usa **curtose** e **negentropia**.

## ICA – Pré-processamento

- Como passos de pré-processamento, os algoritmos usam centragem (subtrair a média para obter um sinal de média zero), branqueamento e redução da dimensionalidade, usualmente via ACP e decomposição em valores singulares. O branqueamento assegura que todas as dimensões são tratadas igualmente antes que o algoritmo seja aplicado.
- Se a  $\text{Cov}(X, Y) = E(XY) - E(X)E(Y) = 0$  dizemos que  $X$  e  $Y$  são não correlacionadas. Se  $X$  e  $Y$  são independentes, então são não correlacionadas. Porém, não correlação não implica em independência, ou seja covariância zero não implica necessariamente em independência, a não ser que  $(X, Y)$  tenham distribuição gaussiana bivariada.
- Uma propriedade um pouco mais forte que não correlação é **brancura** (*whiteness*). Branqueamento de um vetor de média zero,  $\mathbf{y}$ , significa que seus componentes são não correlacionados e suas variâncias são unitárias. Em outras palavras, a matriz de covariância (assim como a matriz de correlação) de  $\mathbf{y}$  é igual a matriz identidade:

$$E[\mathbf{y}\mathbf{y}^\top] = \mathbf{I}. \quad (9)$$

## ICA – Pré-processamento

- Consequentemente, branqueamento significa que transformamos linearmente o vetor de dados observados  $\mathbf{x}$  através de uma multiplicação linear com alguma matriz  $\mathbf{V}$

$$\mathbf{z} = \mathbf{V}\mathbf{x}, \quad (10)$$

obtendo então um novo vetor  $\mathbf{z}$  que é **branco**. O branqueamento é sempre possível. Um método bastante popular é usando a **decomposição em valores singulares** (*singular value decomposition-SVD*) da matriz de covariâncias da variável centrada, digamos  $\tilde{\mathbf{X}}$ :

$$E[\tilde{\mathbf{X}}\tilde{\mathbf{X}}^\top] = \mathbf{E}\mathbf{D}\mathbf{E}^\top, \quad (11)$$

onde  $\mathbf{E}$  é a matriz ortogonal de autovetores de  $E[\tilde{\mathbf{X}}\tilde{\mathbf{X}}^\top]$  e  $\mathbf{D}$  é a matriz diagonal de seus autovalores,  $\mathbf{D} = \text{diag}(d_1, \dots, d_p)$ .

- Branqueamento pode agora ser feito pela matriz de branqueamento

$$\mathbf{V} = \mathbf{E}\mathbf{D}^{-1/2}\mathbf{E}^\top. \quad (12)$$

## ICA – Pré-processamento

- O branqueamento transforma  $\mathbf{A}$  em  $\tilde{\mathbf{A}}$ , tal que

$$\tilde{\mathbf{X}} = \mathbf{ED}^{-1/2} \mathbf{E}^\top \mathbf{A} \mathbf{S} = \tilde{\mathbf{A}} \mathbf{S}.$$

- O procedimento torna  $\tilde{\mathbf{A}}$  ortogonal, pois

$$E[\tilde{\mathbf{X}} \tilde{\mathbf{X}}^\top] = \tilde{\mathbf{A}} E(\mathbf{S} \mathbf{S}^\top) \tilde{\mathbf{A}}^\top = \tilde{\mathbf{A}} \tilde{\mathbf{A}}^\top = \mathbf{I}.$$

- Ainda, o processo de branqueamento reduz o número de parâmetros a estimar, de  $p^2$  para  $p(p - 1)/2$ , devido à ortogonalidade da matriz  $\tilde{\mathbf{A}}$ .

## ICA – Algoritmos

- Há vários algoritmos para a análise de componentes independentes de um conjunto de dados: Algoritmo do Gradiente usando a Curtose, Algoritmo de Ponto Fixo usando Curtose, Algoritmo do Gradiente usando Negentropia, Algoritmo de Ponto Fixo usando Negentropia, Algoritmos para Estimação de Máxima Verossimilhança. Veja Gegembauer (2010) para detalhes.
- Algoritmos tradicionalmente utilizados para a ICA incluem **infomax**, **FastICA** e **JADE**.
- O pacote **ica**, no R, contempla esses algoritmos.
- Outro pacote: **ProDenICA**, este desenvolvido por T. Hastie e R. Tibshirani, que apresenta a particularidade de estimar a densidade de cada componente.

## ICA – Algoritmos

- Um número  $\xi$  diz-se um **ponto fixo** de uma função  $\varphi$  se  $\varphi(\xi) = \xi$ .
- Exemplo:  $\varphi(x) = x^2 - 2$  tem dois pontos fixos,  $-1$  e  $2$
- Encontrar pontos fixos e raízes de polinômios algébricos são problemas equivalentes.
- Se existir um ponto fixo de uma função  $\varphi(x)$  num intervalo  $[a, b]$ , ele pode ser encontrado pelo **algoritmo do ponto fixo**:

$$x^{(k+1)} = \varphi(x^{(k)}), \quad k \geq 0. \quad (13)$$

O algoritmo pode não convergir.

- O método de Newton-Raphson é um algoritmo de ponto fixo.
- **Teorema de convergência do ponto fixo:** Se  $\varphi$  for contínua e  $x^{(k)}$  gerada por (13) e se existir  $\xi$  tal que  $\lim_{k \rightarrow \infty} x^{(k)} = \xi$ , então  $\xi$  é um ponto fixo de  $\varphi$ .

## ICA – Algoritmos

- O pacote **FastICA**, desenvolvido por Hyvärinen and Oja (1997), usa uma aproximação da negrentropia e o algoritmo é baseado em iterações de ponto fixo. É mais robusto que métodos baseados na curtose.
- **JADE**: Joint approximate diagonalization of eigenmatrices. Usa métodos matriciais.
- **InfoMax**: pacote que minimiza a **informação mútua**:

$$I(X, Y) = H(X) - H(X|Y), \quad (14)$$

onde  $H(X|Y)$  é a entropia condicional ( $H(X, Y) - H(Y)$ ) e  $H(X)$  é a entropia de  $X$ .

- No caso contínuo,

$$I(X, Y) = \int f(x, y) \log \frac{f(x, y)}{f_X(x)f_Y(y)} dx dy. \quad (15)$$

- $I(X, Y) = 0$  se, e somente se,  $X$  e  $Y$  independentes.
- Amari et al. (1996): algoritmo InfoMax.

## ICA – Algoritmos

- Uma maneira de extrair as componentes é forçá-las a serem as mais não normais possíveis.
- Negentropia pode ser usada para estimar não-normalidade; é uma medida de distância da normalidade:

$$N(X) = H(X_{\text{normal}}) - H(X). \quad (16)$$

- Para uma dada matriz de covariância, a distribuição que tem a maior entropia é a gaussiana. Assim, a negentropia é uma medida estritamente positiva da não-gaussianidade.
- Como é difícil calcular a negentropia pela definição (16), usa-se a aproximação de Hyvärinen and Oja.

## ICA – Exemplo 1

- **Exemplo 1.** Vamos retomar o exemplo visto na seção de Análise Fatorial, sobre poluição, com 9 variáveis.
- Usaremos o pacote `ica` do R, que tem notação diferente para as diferentes matrizes. Em cada caso faremos a correspondência entre as notações.
- A matriz **M** (*mixing*) estimada, de ordem  $9 \times 9$ , correspondente à matriz **A** do texto, é

```
[,1]          [,2]          [,3]          [,4]          [,5]          [,6]          [,7]          [,8]
[1,] -139.3790190 -149.7021007 -325.3324853 1077.925146 155.3171887 68.243758 111.7946050 -274.782659
[2,] -0.4866963  -0.2949352 -0.5650103  1.190811  0.3524588 0.3555581 0.4229097 -0.879921
[3,] -12.4332654  2.4281842 -25.8117720  42.551213 13.2787874 13.166554 16.9792887 -77.566097
[4,] -42.4416295 -15.1698381 -86.3634728 405.249890 356.4777419 14.649072 -24.5665422 -99.761388
[5,] -264.2307634 -730.1007230 224.9821643 -230.382230 38.0537401 123.889785 9.4797710 87.720764
[6,] -109.9753576 -35.9892578 -184.6838704 661.491067 263.0009705 33.477923 308.4517253 -40.708621
[7,] -121.4505476  21.0283925 -136.7716779 -19.803719 5.3052447 19.311117 15.3286940 2.762610
[8,] -17.9760476 -19.9296505  2.3521578 10.763812 14.0445497 232.900590 -0.7320548 4.559599
[9,] -7362.9405123 3856.0539441 3007.1595557 -1949.521311 -1347.9118353 1263.932257 -977.7231474 -677.758695
[,9]
[1,] 15.571397
[2,] -1.274516
[3,] -8.172252
[4,] -1.558269
[5,] 227.872428
[6,] 57.382165
[7,] 14.380424
[8,] 6.372196
[9,] -609.643712
```

## ICA – Exemplo 1

- Temos, por exemplo,

$$X_1 = -139,379S_1 - 149,702S_2 + \dots + 15,571S_9.$$

- O programa também fornece as matrizes :

$\mathbf{S}$ , de ordem  $193 \times 9$ , matriz estimada das componentes independentes, obtida por

$$\mathbf{S} = \mathbf{W}\mathbf{X} = \mathbf{Y}\mathbf{R};$$

$\mathbf{W}$ , de ordem  $9 \times 9$ , matriz *un-mixing*, tal que  $\mathbf{X}\mathbf{Y}\mathbf{W} = \mathbf{S}$ .  $\mathbf{W}$  é escolhida de modo a maximizar a negrentropia, com a restrição de que  $\mathbf{W}$  seja ortogonal;

$\mathbf{Y}$ , de ordem  $193 \times 9$ , matriz pré-branqueada;

$\mathbf{Q}$ , de ordem  $9 \times 15$ , matriz de pré-branqueamento, tal que  $\mathbf{Y} = \mathbf{Q}\mathbf{X}$ ;

$\mathbf{R}$ , de ordem  $9 \times 9$ ; matriz de rotação ortogonal, tal que  $\mathbf{S} = \mathbf{Y}\mathbf{R}$ .

## ICA – Exemplo 1

- Fornece, também:

vafs : variância explicada por cada c.i.

alg : algoritmo usado (parallel ou deflation)

fun : função contraste (para aproximar a negentropia)

alpha : tuning parameter

iter : número de iterações do algoritmo

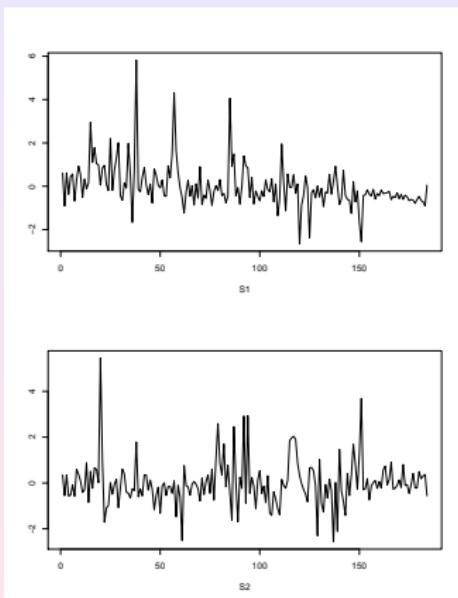
- As variâncias explicadas pelas c.i's são dadas por

0.600985581 0.170645639 0.102434106 0.062162898 0.022555357

0.018513146 0.011778296 0.006196794 0.004728182

- Vemos que as três primeiras componentes explicam cerca de 88% da variância dos dados. Na Figura 1 temos os gráficos das primeiras duas componentes independentes.

# ICA – Exemplo 1



**Figura:** Gráficos das duas primeiras c.i.'s para o Exemplo 13.3.

## ICA – Exemplo 1

A matriz **W** estimada é dada por

```
[,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
[1,] -1.750287e-04 -0.074821293 0.0018890899 -1.219544e-04 -4.394832e-04 -9.771709e-05 -1.951470e-03 9.258294e-04
[2,] -3.688197e-04 -0.193146765 0.0014816133 1.790061e-04 -1.018530e-03 5.109103e-04 -8.073011e-05 5.379454e-04
[3,] -2.673837e-04 -0.024630937 0.0003486251 1.736989e-05 2.822484e-04 6.088192e-04 -5.353500e-03 -1.075091e-04
[4,] 1.176365e-03 -0.020637504 -0.0041025331 -1.933696e-04 -1.798993e-04 -3.679419e-05 -1.609738e-03 5.375905e-05
[5,] -1.511512e-03 0.035410750 0.0012706791 2.872724e-03 1.468919e-04 5.799778e-04 5.234842e-04 1.298423e-05
[6,] -2.234726e-05 -0.004478221 0.0004852519 -1.619936e-04 -1.281089e-04 -1.639626e-05 -1.038226e-04 4.386960e-03
[7,] -1.548268e-03 0.105156093 0.0053742429 -2.024956e-03 9.977071e-05 3.208043e-03 -9.018445e-04 -2.629357e-04
[8,] 1.119943e-04 0.110425897 -0.0148241124 -5.879721e-05 -1.557664e-04 5.604000e-04 1.003537e-03 5.855453e-04
[9,] 3.545273e-04 -0.752508234 0.0074526851 -6.531333e-05 2.558401e-04 4.482189e-04 7.568601e-04 3.558750e-04
[,9]
[1,] -8.288335e-05
[2,] 4.515382e-05
[3,] 7.534598e-05
[4,] 2.056474e-05
[5,] -1.503345e-05
[6,] -3.322196e-06
[7,] -1.102442e-05
[8,] -4.811093e-06
[9,] 1.592838e-06
```

Assim, por exemplo,

$$S_1 = -0,000175X_1 - 0,0748X_2 + \dots + 0,0000829X_9.$$

Vemos que somente as variáveis  $X_3=Zn$ , e  $X_4=Ba$  são as mais importantes para explicar as componentes independentes  $S_i$ ,  $i = 1, \dots, 9$ .

## ICA – Exemplo 1

O pacote ProDenICA do R, fornece as densidades estimadas das c.i.'s, mostradas na Figura 2.

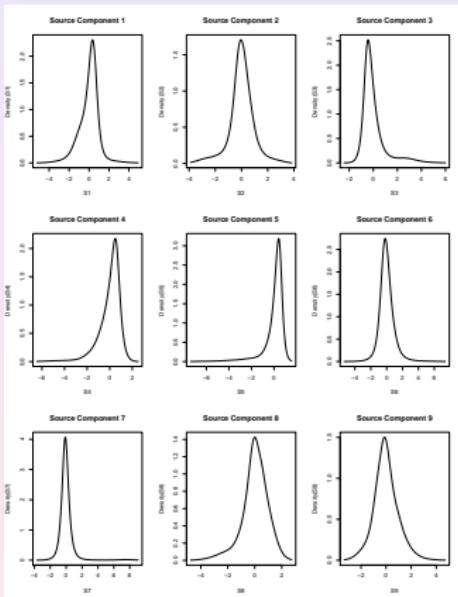


Figura: Gráficos das densidades das c.i.'s para o Exemplo 1.

## ICA – Exemplo 1

Vamos considerar, agora, três componentes independentes. A matriz **A** agora fica

	[,1]	[,2]	[,3]
[1,]	130.0588599	-1172.278514	-50.1191462
[2,]	0.4506871	-1.538268	-0.2837152
[3,]	17.7456231	-64.094123	-17.9385466
[4,]	46.0218183	-489.157909	-30.7345628
[5,]	215.8328265	141.834601	846.2085520
[6,]	85.0325691	-771.625038	-24.1875542
[7,]	76.2477919	-26.575128	-18.3258463
[8,]	61.9279843	-24.349130	49.9983656
[9,]	8353.6861764	3325.603526	-2506.6567435

## ICA – Exemplo 1

- A matriz **W** fica

```
[,1]          [,2]          [,3]          [,4]          [,5]          [,6]          [,7]          [,8]
[1,] 0.0002104413 2.494246e-07 7.925442e-06 8.257729e-05 3.307134e-04 1.416700e-04 1.255409e-05 3.473013e-05
[2,] -0.0005069541 -7.063526e-07 -2.903874e-05 -2.097610e-04 -8.165741e-06 -3.338968e-04 -2.364408e-05 -2.260694e-05
[3,] 0.0000287935 -1.201862e-07 -1.273793e-05 -2.839277e-06 1.091458e-03 2.931156e-05 7.179379e-06 8.385744e-05
[,9]
[1,] 1.056008e-04
[2,] 1.310325e-05
[3,] -2.959094e-05
```

- A variância explicada por cada c.i. é mostrada abaixo:

```
[1] 0.77283297 0.14705258 0.07750351
```

- Vemos que as duas primeiras c.i.'s explicam 92% da variância dos dados. Usando-se a função **fastICA** do pacote **fastICA** do R, podemos fazer alguns gráficos.
- A Figura 3 mostra os dados pré-processados, as componentes principais estimadas e as componentes independentes estimadas. Na Figura 4 temos os gráficos das três componentes.

# ICA – Exemplo 1

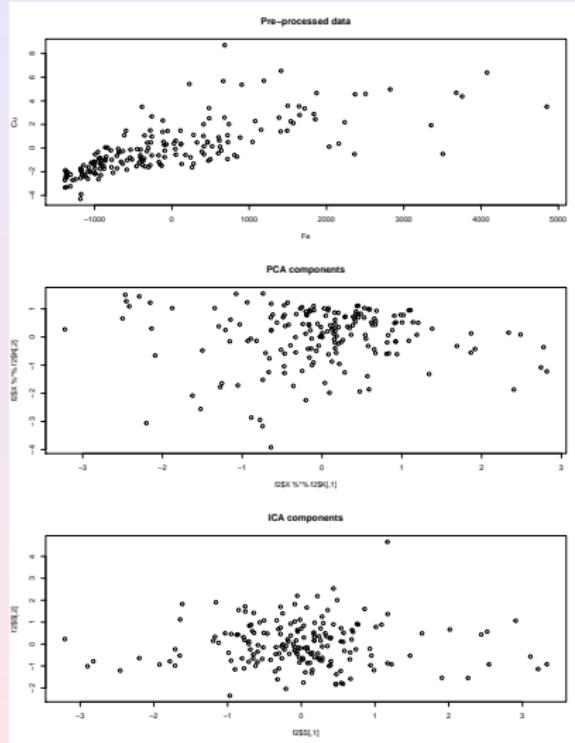
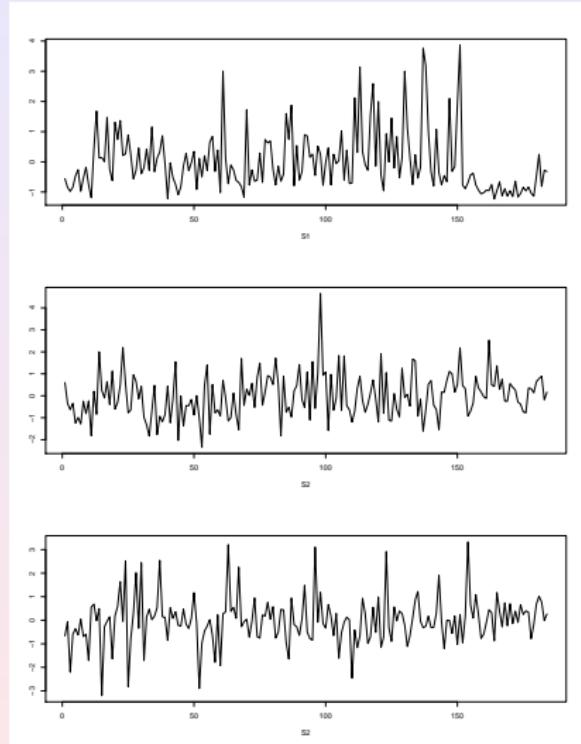


Figura: Gráficos dos dados, CP's e CI's para o Exemplo 1.

# ICA – Exemplo 1



**Figura:** Gráficos das três componentes independentes.

# Decomposição espectral

- Seja  $\mathbf{A}$  uma matriz quadrada de ordem  $m$ . Segue-se que  $|\mathbf{A} - \lambda\mathbf{I}|$  é um polinômio de ordem  $m$  em  $\lambda$  e terá  $m$  raízes complexas  $\lambda_1, \dots, \lambda_m$ . Essas raízes são chamadas *raízes características* ou *autovalores* de  $\mathbf{A}$ .
- Como  $\mathbf{A} - \lambda_j\mathbf{I}$  é singular,  $j = 1, \dots, m$ , existe um vetor  $\mathbf{a}_j$ , cujas coordenadas não são todas nulas, tal que  $(\mathbf{A} - \lambda_j\mathbf{I})\mathbf{a}_j = \mathbf{0}$ , ou seja,  $\mathbf{A}\mathbf{a}_j = \lambda_j\mathbf{a}_j$ ,  $j = 1, \dots, m$ . Os vetores  $\mathbf{a}_1, \dots, \mathbf{a}_m$  são chamados *vetores característicos* ou *autovetores* de  $\mathbf{A}$ .
- Os seguintes resultados são válidos.
  - (1) O posto de  $\mathbf{A}$ ,  $\rho(\mathbf{A})$ , dá o número de autovalores de  $\mathbf{A}$  não nulos.
  - (2)  $\text{tr}(\mathbf{A}) = \sum_{j=1}^m \lambda_j$ .
  - (3)  $|\mathbf{A}| = \prod_{j=1}^m \lambda_j$ .
  - (4) Se  $\mathbf{A}$  é uma matriz simétrica, real, todos os seus autovalores são reais, e para cada autovalor real existe um autovetor real.
  - (5) Se  $\mathbf{A}$  é simétrica, real, os autovetores correspondentes a autovalores distintos são ortogonais.
  - (6) Se  $\mathbf{A}$  é não negativa definida, então  $\lambda_j \geq 0$ ,  $j = 1, \dots, m$ .
  - (7) Se  $\mathbf{A}$  é simétrica, de ordem  $m \times m$ , existe uma matriz ortogonal  $\mathbf{X}$ , tal que

$$\mathbf{X}' \mathbf{A} \mathbf{X} = \Lambda = \text{diag}\{\lambda_1, \dots, \lambda_m\},$$

ou  $\mathbf{A} = \mathbf{X} \Lambda \mathbf{X}'$ , onde os  $\lambda_j$  são os autovalores de  $\mathbf{A}$  e as colunas de  $\mathbf{X}$  são os correspondentes autovetores.

## Decomposição espectral

- O resultado (7) é chamado o *teorema espectral* para matrizes simétricas. Segue-se que a *decomposição espectral* de  $\mathbf{A}$  é dada por

$$\mathbf{A} = \sum_{j=1}^m \lambda_j \mathbf{x}_j \mathbf{x}_j'$$

onde  $\mathbf{x}_j$  é o autovetor correspondente a  $\lambda_j$ .

- Se  $\mathbf{A}$  é uma matriz quadrada de ordem  $m$ , positiva definida, existe uma matriz triangular inferior  $\mathbf{T}$ , com elementos da diagonal principal positivos, tal que

$$\mathbf{T}^{-1} \mathbf{A} (\mathbf{T}')^{-1} = \mathbf{I}_m, \quad \text{ou} \quad \mathbf{A} = \mathbf{T} \mathbf{T}'.$$

- Esta decomposição de  $\mathbf{A}$  é chamada *decomposição de Cholesky*.

## Decomposição em valores singulares - DVS

- DVS é uma fatoração de uma matriz qualquer em outras três matrizes com características importantes.
- Dada uma matriz  $\mathbf{A}$ , de ordem  $m \times n$ ,  $m \geq n$ , não necessariamente de posto máximo, uma DVS de  $\mathbf{A}$  é uma fatoração da forma  $A = \mathbf{U}\Sigma\mathbf{V}^\top$ , onde:
  - $\mathbf{U}$ , de ordem  $m \times n$ , é ortogonal;
  - $\mathbf{V}$ , de ordem  $m \times n$ , é ortogonal;
  - $\Sigma$  é diagonal se  $m = n$ , caso contrário adicionam-se  $m - n$  linhas de zeros em  $\Sigma$ .
  - Os valores  $\sigma_1, \dots, \sigma_n$  da diagonal principal de  $\Sigma$  são chamados **valores singulares** de  $\Sigma$  e  $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_n > 0$ .
  - As colunas de  $\mathbf{U}$ ,  $u_1, \dots, u_m$  são chamados **vetores singulares à esquerda** de  $\mathbf{A}$ , enquanto as colunas de  $\mathbf{V}$ ,  $v_1, \dots, v_n$  são chamados **vetores singulares à direita** de  $\mathbf{A}$ , de tal forma que  $\mathbf{A}v_j = \sigma_j u_j$ .

## Decomposição em valores singulares - DVS

- Decomposição por Autovalores  $\mathbf{A} = \mathbf{X}\Lambda\mathbf{X}^{-1}$ ;
  - usa a mesma base  $\mathbf{X}$  para os espaços linha e coluna;
  - geralmente não usa base ortonormal;
  - é definida somente para matrizes quadradas.
- DVS
  - usa bases diferentes  $\mathbf{V}$ ,  $\mathbf{U}$  para os espaços linha e coluna;
  - usa bases ortonormais;
  - existe para todas as matrizes.
- Entretanto, para matrizes simétricas positivas definidas, a decomposição por autovalores e a DVS são idênticas.

# Aplicações da DVS

- Cálculo de propriedades de matrizes;
- Aproximação de  $\mathbf{A}$  por matrizes de posto baixo;
- Processamento de Sinais e Imagens
  - Compressão de Imagens;
  - Eliminação de ruídos (ruídos normalmente possuem  $\sigma_j$  pequenos);
  - Recuperação de Informações.

## Curtose

- 1) Seja  $X$  uma variável aleatória qualquer, com média  $\mu$  e variância  $\sigma^2$ . Então, a assimetria de  $X$  é definida por

$$A(X) = E \left( \frac{(X - \mu)^3}{\sigma^3} \right), \quad (17)$$

enquanto a curtose de  $X$  é definida por

$$K(X) = E \left( \frac{(X - \mu)^4}{\sigma^4} \right). \quad (18)$$

- 2) Para uma distribuição normal,  $A = 0$  e  $K = 3$ , donde a quantidade  $e(X) = K(X) - 3$  ser chamada excesso de curtose. Distribuições com caudas pesadas têm curtose maior do que 3, e esta pode mesmo ser infinita.

## Curtose

- 3) Com uma amostra  $X_1, \dots, X_T$  de  $X$ , considere o  $r$ -ésimo momento amostral

$$m_r = \frac{1}{T} \sum_{t=1}^T (X_t - \bar{X})^r,$$

onde  $\hat{\mu} = \bar{X}$ . Substituindo os momentos verdadeiros de  $X$  pelos respectivos momentos amostrais, obtemos os estimadores

$$\hat{A}(X) = \frac{m_3}{m_2^{3/2}} = \frac{1}{T} \sum_{t=1}^T \left( \frac{X_t - \bar{X}}{\hat{\sigma}} \right)^3, \quad (19)$$

$$\hat{K}(X) = \frac{m_4}{m_2^2} = \frac{1}{T} \sum_{t=1}^T \left( \frac{X_t - \bar{X}}{\hat{\sigma}} \right)^4, \quad (20)$$

respectivamente, onde  $\hat{\sigma}^2 = \sum_{t=1}^T (X_t - \bar{X})^2 / T$ . Segue-se que  $\hat{e}(X) = \hat{K}(X) - 3$ .

- 4) Pode-se provar que, se tivermos uma amostra de uma distribuição normal, e  $T$  for grande, então

$$\hat{A} \sim \mathcal{N}(0, 6/T), \quad \hat{K} \sim \mathcal{N}(3, 24/T). \quad (21)$$

Esses fatos podem ser utilizados para testar a normalidade de uma série (Jarque-Bera).

## Referências

- Amari, S., Cichocki, A. and Yang, H. H. (1996). A new learning algorithm for blind signal separation. *Advances in Neural Information Processing Systems*, **8**, 757–763.
- Gegembauer, H. V. (2010). *Análise de Componentes Independentes com Aplicações em Séries Temporais Financeiras*. Dissertação de Mestrado, IME-USP.
- Hyvärinen, A. and Oja, E. (1997). A fast fixed point algorithm for independent component analysis. *Neural Computation*, **9**, 1483–1492.
- Hyvärinen, A. (1999). Fast and robust fixed-point algorithm for independent component analysis. *IEEE Transactions on Neural Network*, **10**, 626–634.
- Hyvärinen, A. and Oja, E. (2000). Independent component analysis: Algorithms and applications. *Neural Networks*, **13**, 411–430.
- Hyvärinen, A., Karhunen, J. and Oja, E. (2001). *Independent Component Analysis*. New York: Wiley.

# MAE 5905: Introdução à Ciência de Dados

Pedro A. Morettin

Instituto de Matemática e Estatística  
Universidade de São Paulo  
pam@ime.usp.br  
<http://www.ime.usp.br/~pam>

## Aula 16 - Parte 1

13 de junho de 2023

# Sumário

1 Redes Neurais

2 Perceptron

3 RN com uma camada

## Preliminares

- Uma **rede neural** (RN) é um conjunto de algoritmos construídos para identificar relações entre as variáveis de um conjunto de dados por intermédio de um processo que tenta imitar a maneira com que neurônios interagem no cérebro.
- Cada neurônio numa rede neural é uma função à qual dados são alimentados e transformados numa resposta como em modelos de regressão. Essa resposta é repassada para um ou mais neurônios da rede que ao final do processo produz um resultado ou recomendação.
- Quando falamos em arquitetura de uma RN, estamos nos referindo à maneira como os neurônios estão organizados em camadas. Há três classes diferentes de arquiteturas :
  1. **RN com uma única camada**. Nessa classe, há uma camada de entrada e uma de saída. Essa RN é do tipo **feedforward**. Essa rede também é chamada **perceptron**.
  2. **RN multicamadas**, também do tipo **feedforward**. São redes com uma ou mais camadas escondidas. Cada neurônio de uma camada tem como entradas os neurônios da camada precedente.
  3. **RN recorrentes**. Nesse caso, pelo menos um neurônio conecta-se com um neurônio da camada precedente, criando um **ciclo** (**feedback**).

## História das RN

- As contribuições pioneiras para a área de Redes Neurais (também denominadas redes neurais, termo derivado de neurônio) foram as de McCulloch e Pitts (1943), que introduziram a unidade lógica com limiar (*thresholded logic unit*) e de Hebb (1949), que postulou a primeira regra para aprendizado organizado.
- Rosenblatt (1957) introduziu o **perceptron** e Widrow e Hoff (1960) o **adaline** (*adaptive linear neuron*).
- Depois, seguiu-se o que foi chamado de **primeiro inverno neural**. Minsky e Papert (1969) escreveram um livro, chamado *Perceptron*, em que apresentam o problema **XOR** (*exclusive OR*). O problema consistia em usar redes neurais (o perceptron) para prever saídas da lógica XOR, dadas duas entradas binárias, como na Tabela 1. Uma função XOR deve retornar um valor verdadeiro (1, ponto verde na figura) se as duas entradas não são iguais e um valor falso (0, ponto vermelho na Figura 1) se elas são iguais.

## História das RN

Tabela 1: Problema XOR

Entrada 1	Entrada 2	Saída
0	0	0
0	1	1
1	1	0
1	0	1

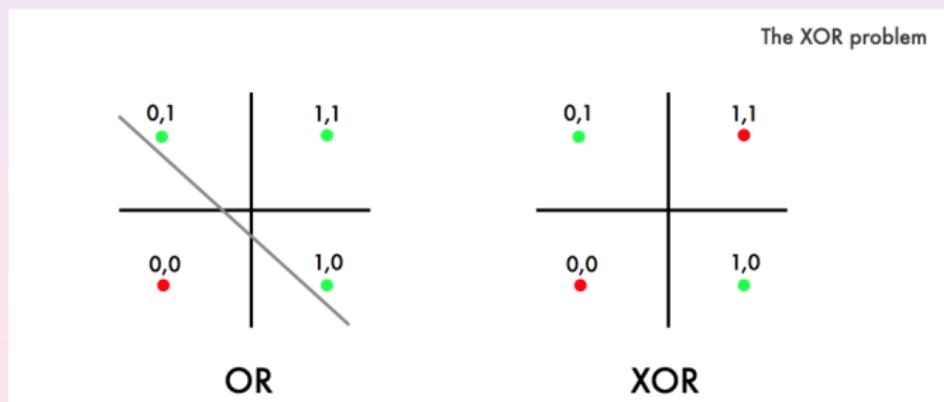


Figura 1: Os problemas lógicos (a) OR e (b) XOR.

# História das RN

- Esse é um problema de classificação usando o perceptron, mas este supõe que as duas classes sejam linearmente separáveis, o que elas não são, no caso XOR, conforme a Figura 1. O problema OR (inclusive) é separável por uma reta.
- Este problema seria solucionado com a introdução, nas décadas de 1980 e 1990, das redes com várias camadas ocultas, o algoritmo de retroalimentação (*backpropagation*) e as redes neurais convolucionais. Veja LeCun (2015) para detalhes.
- Um **segundo inverno neural** ocorre na década de 1990, no qual a contribuição mais significativa, fora do contexto de redes neurais, foi a de Cortes e Vapnik (1995), que introduziram as máquinas de suporte vetorial (*support vector machines*).

# História das RN

- A partir de 2006 foram desenvolvidas as **redes neurais profundas** (*deep neural networks*, DNN), com menção especial às **redes neurais recorrentes** (RNN) e as **redes neurais convolucionais** (CNN). É a chamada **era das GPU** (Graphic Processing Units).
- Em especial, destacamos uma competição promovida pela ImageNet, que mantém uma base de dados com milhões de imagens para fins de treinamento e classificação, usando técnicas de ciências de dados. Krizhevsky et al. (2012) venceram essa competição em 2012, usando uma CNN com sete camadas ocultas, denominada **Alex net** (primeiro nome do primeiro autor) e que levou seis dias para ser treinada. A proporção de erros dessa rede foi de 15,3%. Em 2015, o vencedor usou uma CNN com 150 camadas ocultas e a proporção de erros foi de 3,5%.
- A seguir faremos uma revisão dessas redes neurais.

## Perceptron

- Rosenblatt (1958) introduziu o algoritmo *perceptron* como o primeiro modelo de aprendizado supervisionado. A ideia do algoritmo é atribuir pesos  $w_i$  aos dados de entrada  $\mathbf{x}$ , iterativamente, até que o processo tenha uma precisão pré-especificada para a tomada de decisão. No caso de classificação binária, o objetivo é classificar elementos do conjunto de dados com valor das variáveis preditoras  $\mathbf{x}$  (dados de entrada) em uma de duas classes, rotuladas aqui por +1 e -1.
- O algoritmo *perceptron* (programado para um computador IBM 704) foi implementado em uma máquina chamada Mark I, planejada para reconhecimento de imagens. O modelo subjacente consiste de uma combinação linear das entradas,  $\mathbf{x}$ , com a incorporação de um viés externo, cujo resultado é comparado com um limiar, definido por meio de uma **função de ativação**. As funções de ativação usualmente empregadas são as funções **degrau** ou **sigmoide**. Outras funções de ativação serão vistas a seguir.
- Se  $\mathbf{x} = (1, x_1, x_2, \dots, x_p)^\top$  contém as entradas,  $\mathbf{w} = (-b, w_1, w_2, \dots, w_p)^\top$  são os pesos, a saída é dada por

$$v = \sum_{i=0}^p w_i x_i = \mathbf{w}^\top \mathbf{x}.$$

# Perceptron

- Definamos uma **função de ativação**  $f(v)$  tal que se  $f(v) \geq b$ , a amostra é classificada na classe  $+1$  e se  $f(v) < b$ , é classificada na classe  $-1$ . Ou seja,

$$f(v) = \begin{cases} 1, & \text{se } v \geq b \\ -1, & \text{se } v < b. \end{cases}$$

- O parâmetro  $b$  é chamado de **viés**. A ideia de Rosenblatt era obter um algoritmo (regra de aprendizado), segundo a qual os pesos são atualizados a fim de se obter uma fronteira de decisão **linear**, que permite discriminar entre as duas classes linearmente separáveis. Veja a Figura 2.

# Perceptron

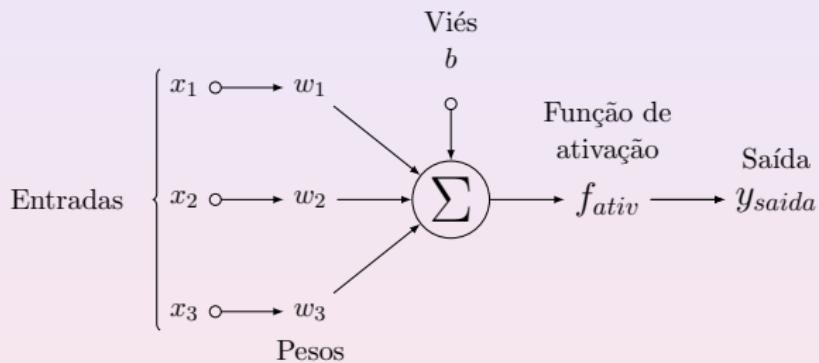


Figura 2: Diagrama de um perceptron.

## Algoritmo do perceptron

- [1] Inicialize todos os pesos como sendo zero ou valores aleatórios pequenos;
- [2] Para cada amostra de treinamento  $\mathbf{x}^{(i)}$ :
  - (a) calcule os valores da saída;
  - (b) atualize os pesos.

A atualização dos pesos é feita segundo a regra de aprendizagem

$$\Delta w_j = \eta(\text{alvo}^{(i)} - \text{saída}^{(i)})x_j^{(i)},$$

onde  $\eta$  é a taxa de aprendizagem (um valor entre 0 e 1), **alvo** é o rótulo verdadeiro da classe e **saída** é o rótulo da classe prevista. Todos os pesos são atualizados simultaneamente. Por exemplo, no caso de duas variáveis,  $x_1$  e  $x_2$ , teremos que atualizar  $w_0$ ,  $w_1$  e  $w_2$ .

## Algoritmo do perceptron

- É fácil ver que nos dois casos para os quais o perceptron prevê o rótulo da classe verdadeira,  $\Delta w_j = 0$ , para todo  $j$ . No caso de previsão errônea,  $\Delta w_j = 2\eta x_j^{(i)}$  ou  $\Delta w_j = -2\eta x_j^{(i)}$ .
- Uma observação importante é que a convergência do perceptron somente é garantida se as duas classes são linearmente separáveis. Se não forem, podemos fixar um número máximo de passadas sobre os dados de treinamento (**épocas**) ou um limiar para o número máximo de classificações erradas tolerável.

## Perceptron – Exemplo 1

**Exemplo 1.** Consideremos um exemplo simulado. Simulamos 50 valores de  $X_1$  e  $X_2$ , geradas segundo uma normal padrão e  $Y$  será igual a +1 se  $X_2 > 0,5X_1 + 0,2$  e igual a -1, caso contrário. Veja a Figura 3.

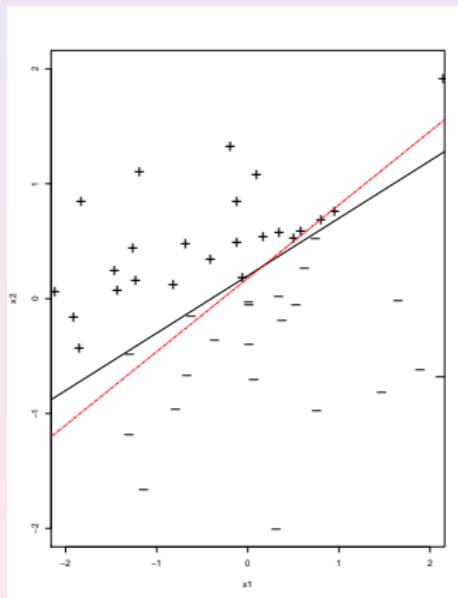


Figura 3. Exemplo 1: reta simulada (preto); perceptron (vermelho).

## Perceptron – Exemplo 1

Usando o *script* para o perceptron (veja a página do livro), obtemos as estimativas dos coeficientes

\$w

x1	x2
-0.5384671	0.8426465

\$b

[1] -0.1495257

e, portanto a reta  $x_2 = 0,17745 + 0,639x_1$ , que está apresentada também na Figura 2. Ambas as retas estão próximas.

## Perceptron – Exemplo 2

**Exemplo 2.** Vamos considerar o conjunto de dados [Iris](#) e duas variáveis,

$x_1$ : comprimento de sépala

$x_2$ : comprimento de pétala

e duas espécies, Iris versicolor (+1) e Iris setosa (-1). Veja a Figura 4.

Usando, novamente, o *script* para o perceptron, obtemos

\$w

```
sepal      petal
1 0.1419446 0.9898746
```

\$b

```
[1] -2.855304
```

de modo que obtemos a reta  $x_2 = 2,8844 - 0,14335x_1$ , representada na Figura 4.

## Perceptron – Exemplo 2

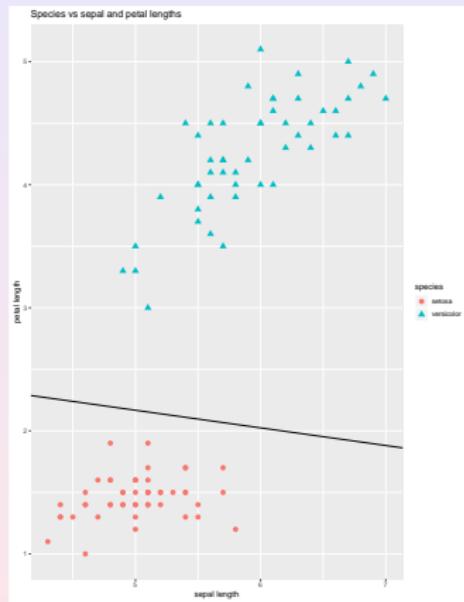


Figura 4. Iris data com duas variáveis e reta perceptron.

## RN com uma camada

Atualmente, a RN mais simples consiste de entradas, de uma camada intermediária escondida e das saídas. Na Figura 5, se  $K = 1$  temos o caso de regressão e, no caso de classificação, teremos  $K$  classes, sendo que  $Z_i = -1$  ou  $Z_i = +1$ .

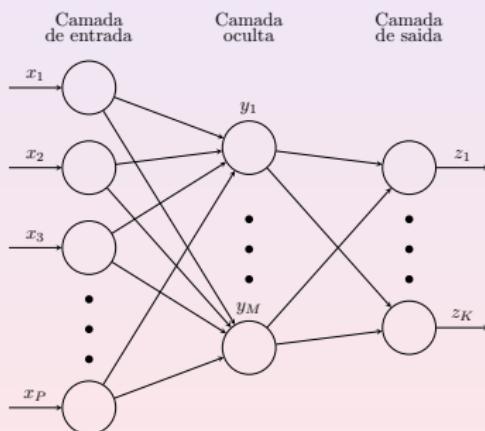


Figura 5. Rede neural com uma camada escondida.

## RN com uma camada

- Consideremos os seguintes vetores de variáveis,  
 $\mathbf{X} = (X_1, \dots, X_p)^\top$ ,  $\mathbf{Y} = (Y_1, \dots, Y_M)^\top$  e  $\mathbf{Z} = (Z_1, \dots, Z_K)^\top$  e sejam, os vetores de pesos  $\boldsymbol{\alpha}_j$ ,  $j = 1, \dots, M$ ,  $\boldsymbol{\beta}_k$ ,  $k = 1, \dots, K$ , de ordens  $p \times 1$  e  $M \times 1$ , respectivamente.
- A RN da Figura 5 pode ser representada pelas equações:

$$Y_j = h(\alpha_{0j} + \boldsymbol{\alpha}_j^\top \mathbf{X}), \quad j = 1, \dots, M, \quad (1)$$

$$Z_k = g(\beta_{0k} + \boldsymbol{\beta}_k^\top \mathbf{Y}), \quad k = 1, \dots, K. \quad (2)$$

- As funções  $h$  e  $g$  são chamadas **funções de ativação** e geralmente são usadas as seguintes funções:

## RN com uma camada

a) logística (ou sigmoide),

$$f(x) = \frac{1}{1 + e^{-x}},$$

b) tangente hiperbólica,

$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}},$$

c) ReLU (rectified linear unit),

$$f(x) = \max(0, x).$$

d) Uma modificação da ReLU é *leaky ReLU*, dada por

$$f(x) = \begin{cases} x, & \text{se } x > 0, \\ 0,01x, & \text{se } x < 0. \end{cases}$$

## Funções de ativação

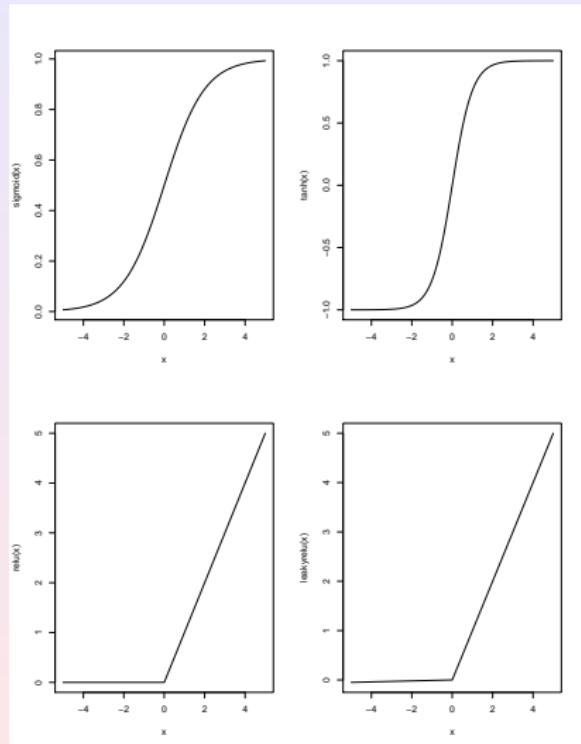


Figura 6. Algumas funções de ativação.

## RN com uma camada

- A função (d) é bastante utilizada, pois é fácil de otimizar, o gradiente pode ser facilmente calculado e converge mais rápido do que sigmoides. Todavia, ela não é derivável na origem, e no algoritmo **backpropagation**, por exemplo, necessitamos de derivabilidade.
- Pode-se usar a mesma função de ativação em (1) e (2). Também é comum que haja uma saída única,  $Z$ , na Figura 5.
- Os pesos  $\alpha_{0j}$  e  $\beta_{0k}$  têm o mesmo papel de  $b$  no perceptron e representam vieses. Os  $Y_j$  constituem a camada escondida e não são observáveis.
- No lugar de (1)–(2), podemos considerar a saída da RN expressa na forma

$$f(\mathbf{x}, \mathbf{w}) = \varphi \left( \sum_{j=0}^{M-1} w_j \phi_j(\mathbf{x}) \right), \quad (3)$$

onde  $\varphi(\cdot)$  é a identidade, no caso de regressão e uma função não linear, no caso de classificação;  $w_j$  são pesos a serem determinados.

## RN com uma camada

Com essa formulação, os seguintes passos são usualmente utilizados na análise de redes neurais (RN) (Bishop, 2006):

- (i) Considere as ativações

$$a_j = \sum_{i=0}^p w_{ji}^{(1)} x_i, \quad j = 1, \dots, M, \quad (4)$$

onde incluímos os vieses  $w_{j0}^{(1)}$  nos pesos  $\mathbf{w}_j^{(1)} = (w_{j0}, w_{j1}, \dots, w_{jp})^\top$ ,  $j = 1, \dots, M$ , fazendo  $x_0 = 1$ . O índice (1) indica a primeira camada da RN.

- (ii) Cada ativação  $a_j$  é transformada usando uma função de ativação  $h(\cdot)$ , resultando

$$y_j = h(a_j). \quad (5)$$

Aqui, podemos usar uma das funções acima. Dizemos que os  $y_j$  são as unidades escondidas.

## RN com uma camada

(iii) Considere as ativações de saída

$$a_k = \sum_{j=0}^M w_{kj}^{(2)} z_j, \quad k = 1, 2, \dots, K, \quad (6)$$

onde, novamente incluimos os vieses  $w_{k0}^{(2)}$  no vetor  $\mathbf{W}$ .

(iv) Finalmente, essas ativações são transformadas por meio de uma nova função de ativação, resultando nas saídas  $z_k$  da RN, sendo que no caso de regressão,  $z_k = a_k$  e no caso de classificação,

$$Z_k = \sigma(a_k), \quad (7)$$

sendo que  $\sigma(a)$  em geral é a logística dada em (a) acima.

## RN com uma camada

- Combinando, obtemos

$$f_k(\mathbf{x}, \mathbf{W}) = \sigma \left( \sum_{j=0}^M w_{kj}^{(2)} h \left( \sum_{i=0}^p w_{ji}^{(1)} x_i \right) \right). \quad (8)$$

- O procedimento para obter (8) é chamado **forward propagation** da informação.
- Podemos generalizar considerando camadas adicionais.
- A nomenclatura de tal rede difere segundo autores e pacotes computacionais. Pode ser chamada de RN com 3 camadas ou uma RN com uma camada escondida. Bishop (2006) sugere chamá-la de uma RN com duas camadas, referindo-se aos pesos  $w_{ji}^{(1)}$  e  $w_{kj}^{(2)}$ . Ainda, o pacote **neuralnet** estipula o número de neurônios,  $M$ , na camada escondida.

## RN com uma camada–Exemplo 3

- **Exemplo 3.** Vamos considerar os dados do Exemplo 1 e vamos criar uma rede neuronal com uma camada escondida com 3 neurônios e função de ativação tangente hiperbólica. Para tanto, usamos o pacote **neuralnet** do R.
- ```
nn=neuralnet(y~x1+x2,data=df, hidden=3, act.fct="tanh",
+linear.output=FALSE)# hidden=3: single layer with 3 neurons
plot(nn)
```
- Obtemos a Figura 7. Nesse exemplo,  $p = 2$ ,  $M = 3$  e  $K = 1$ . As equações (3)–(4) ficam:

$$\begin{aligned}Y_1 &= -4,179 - 2,761X_1 - 19,010X_2 \\Y_2 &= 2,650 - 2,109X_1 - 0,978X_2 \\Y_3 &= 0,582 + 1,472X_1 - 1,232X_2.\end{aligned}$$

- Também, teremos três vetores  $\alpha_1, \alpha_2$  e  $\alpha_3$ , de ordens  $2 \times 1$ ,  $\mathbf{X} = (X_1, X_2)$ ,  $\mathbf{Y} = (Y_1, Y_2, Y_3)$  e  $Z = g_1(W) = W = \beta_{01} + \beta_1^\top \mathbf{Y}$ , com  $\beta_1 = (\beta_1, \beta_2, \beta_3)$ ,  $\beta_{01} = -0,540$ . Ou seja,

$$Z = -0,540 - 3,019Y_1 - 3,538Y_2 - 21,641Y_3.$$

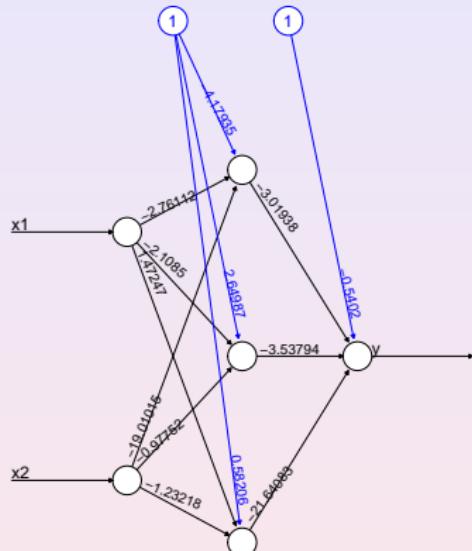
## RN com uma camada—Exemplo 3

Os pesos finais são dados por:

```
result.matrix
```

|                       | [ ,1 ]        |
|-----------------------|---------------|
| error                 | 0.003675542   |
| reached.threshold     | 0.007275895   |
| steps                 | 217.000000000 |
| Intercept.to.1layhid1 | -4.179347633  |
| x1.to.1layhid1        | -2.761123538  |
| x2.to.1layhid1        | -19.010153927 |
| Intercept.to.1layhid2 | 2.649870079   |
| x1.to.1layhid2        | -2.108496830  |
| x2.to.1layhid2        | -0.977518529  |
| Intercept.to.1layhid3 | 0.582063062   |
| x1.to.1layhid3        | 1.472465125   |
| x2.to.1layhid3        | -1.232181252  |
| Intercept.to.y        | -0.540201763  |
| 1layhid1.to.y         | -3.019378642  |
| 1layhid2.to.y         | -3.537936157  |
| 1layhid3.to.y         | -21.640828433 |

## RN com uma camada—Exemplo 3



Error: 0.003676 Steps: 217

Figura 7. Rede neural com uma camada escondida para o Exemplo 3.

## RN com uma camada—Exemplo 3

Vamos ver como fazer previsões para um conjunto de teste. Suponha que tenhamos 5 observações de  $x$ , a saber

| $x_1$      | $x_2$       |
|------------|-------------|
| 0.6180602  | 0.14261227  |
| 0.5053168  | 1.48586481  |
| 0.3615404  | -0.04880704 |
| -0.1707795 | -1.09666571 |
| -0.6284641 | -0.54107065 |

que chamaremos de *test*. Usamos os comandos

```
test=data.frame(x1,x2)
```

```
Predict=compute(nn,test)
```

```
Predict$net.result
```

```
[,1]
[1,] -1.0000000
[2,] 1.0000000
[3,] 1.0000000
[4,] -0.9993473
[5,] -0.9997057
```

## RN com uma camada–Exemplo 3

Convert probabilities into binary classes setting threshold level =0.5

```
prob<-Predict\$/net.result  
pred<-ifelse(prob>0.5,1,0)  
pred
```

```
[,1]  
[1,] 0  
[2,] 1  
[3,] 1  
[4,] 0  
[5,] 0
```

onde agora 0 corresponde a -1 e 1 a +1. Ou seja, classificamos a segunda e terceira observações testes na classe +1 e a primeira, quarta e quinta na classe -1.

## RN com uma camada–Exemplo 4

- **Exemplo 14.4.** Vamos considerar novamente o conjunto Iris, mas somente a espécie Setosa e dois preditores, Petal.Length e Petal.Width. Para uma rede neuronal com três neurônios na camada escondida, obtemos a Figura 8.
- Os pesos obtidos estão indicados abaixo, e vemos que foram necessárias 41 iterações, o limiar foi 0,00994 e o erro de classificação 0,0125.

## RN com uma camada—Exemplo 4

|                                  |              |
|----------------------------------|--------------|
| error                            | 0.012533834  |
| reached.threshold                | 0.009935566  |
| steps                            | 41.000000000 |
| Intercept.to.1layhid1            | -5.266797930 |
| Petal.Length.to.1layhid1         | 1.029448892  |
| Petal.Width.to.1layhid1          | 4.035261475  |
| Intercept.to.1layhid2            | 4.783018629  |
| Petal.Length.to.1layhid2         | -1.313269314 |
| Petal.Width.to.1layhid2          | -2.377663815 |
| Intercept.to.1layhid3            | 3.230647942  |
| Petal.Length.to.1layhid3         | -0.630478727 |
| Petal.Width.to.1layhid3          | -2.790017234 |
| Intercept.to.Species == "setosa" | -1.171205492 |
| 1layhid1.to.Species == "setosa"  | -3.662576010 |
| 1layhid2.to.Species == "setosa"  | 3.664317725  |
| 1layhid3.to.Species == "setosa"  | 3.064883525  |

## RN com uma camada—Exemplo 4

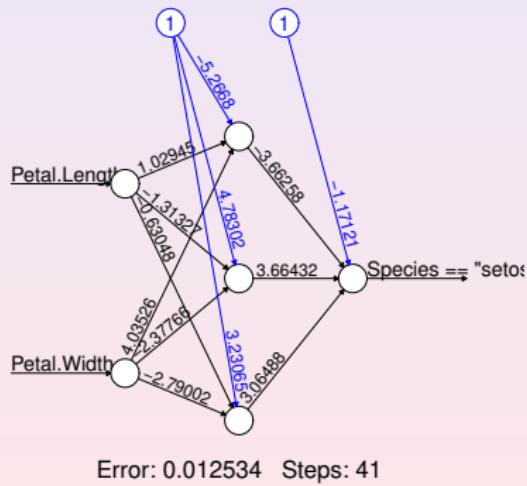


Figura 8. Rede neural com uma camada escondida para o Exemplo 4.

## O algoritmo backpropagation

- O ajuste de modelos de RN é feito minimizando uma função perda, usualmente a soma dos quadrados dos resíduos, no caso de regressão, onde a minimização é feita sobre os pesos. No caso de classificação, usamos a entropia. Nos dois casos é usado um algoritmo chamado de **backpropagation** (BP).
- O algoritmo BP calcula o gradiente da função perda com respeito aos pesos da rede, atualizando para trás uma camada de cada vez, a fim de minimizar a perda. Comumente usa-se **gradient descent** ou variações, como **stochastic gradient descent**.
- É necessário escolher valores iniciais e regularização (usando uma função penalizadora), porque o algoritmo de otimização é não convexo e instável.

# O algoritmo backpropagation

De modo geral, o problema de otimização da RN pode ser posto na forma:

$$\hat{\mathbf{w}} = \arg \min_{\mathbf{w}} \tilde{Q}_n(\mathbf{w}) = \arg \min_{\mathbf{w}} [\lambda_1 Q_n(\mathbf{w}) + \lambda_2 Q^*(\mathbf{w})], \quad (9)$$

com  $\lambda_1, \lambda_2 > 0$ , sendo

$$Q_n(\mathbf{w}) = \sum_{i=1}^n [y_i - f(\mathbf{x}_i, \mathbf{w})]^2, \quad (10)$$

e  $Q^*(\mathbf{w})$  um termo de regularização, que pode ser escolhido entre aqueles estudados no Capítulo 6 (lasso, ridge ou elastic net).

## O algoritmo backpropagation

- Podemos pensar uma rede neural como em (3), ou seja, uma função não linear paramétrica (determinística) de uma entrada  $\mathbf{x}$ , tendo  $\mathbf{z}$  como saída.
- Suponha que tenhamos os vetores de treinamento  $\mathbf{x}_i$ , e os vetores alvos (saídas)  $\mathbf{z}_i$ ,  $i = 1, \dots, n$  e considere a soma dos quadrados dos erros (10), ligeiramente modificada

$$Q_n(\mathbf{w}) = \frac{1}{2} \sum_{i=1}^n \|z_i - f(\mathbf{x}_i, \mathbf{w})\|^2, \quad (11)$$

que queremos minimizar.

# O algoritmo backpropagation

[1] Vamos tratar primeiramente o problema de **regressão** e considerar a rede agora com um erro aleatório  $\varepsilon_i$  acrescentado antes da saída, de modo que agora temos um modelo probabilístico. Por simplicidade, consideremos a saída  $\mathbf{z} = (z_1, \dots, z_n)^\top$  e os erros com distribuição normal, com média zero e variância  $\sigma^2$ , de modo que

$$\mathbf{z} \sim N_n(f(\mathbf{x}_i, \mathbf{w}), \sigma^2 \mathbf{I}).$$

No caso em questão, a função de ativação de saída é a identidade. Chamando  $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_n]$ , a verossimilhança pode ser escrita como

$$L(\mathbf{z}|\mathbf{X}, \mathbf{w}, \sigma^2) = \prod_{i=1}^n p(z_i|\mathbf{x}_i, \mathbf{w}, \sigma^2).$$

Maximizar a log-verossimilhança é equivalente a minimizar (10), de modo que obtemos  $\hat{\mathbf{w}}_{MV}$ . Encontrado  $\hat{\mathbf{w}}_{MV}$ , podemos obter o estimador de MV de  $\sigma^2$ . Como temos uma função não linear,  $Q_n(\mathbf{w})$  é não convexa, portanto na prática podemos obter máximos não locais da verossimilhança.

# O algoritmo backpropagation

(2) No caso de **classificação binária**, para a qual, por exemplo,  $z = +1$  indica a classe  $C_1$  e  $z = 0$  indica a classe  $C_2$ , considere a RN com saída única  $z$  com função de ativação logística,

$$z = \sigma(a) = \frac{1}{1 + e^{-a}},$$

de modo que  $0 \leq f(\mathbf{x}, \mathbf{w}) \leq 1$ . Podemos interpretar  $f(\mathbf{x}, \mathbf{w}) = P(C_1|\mathbf{x})$  e  $P(C_2|\mathbf{x}) = 1 - f(\mathbf{x}, \mathbf{w})$ . Segue que a distribuição de  $z$ , dado  $\mathbf{x}$ , é dada por

$$f(z|\mathbf{x}, \mathbf{w}) = f(\mathbf{x}, \mathbf{w})^z [1 - f(\mathbf{x}, \mathbf{w})]^{1-z}. \quad (12)$$

## O algoritmo backpropagation

- Considerando as observações de treinamento i.i.d., a função erro será dada pela entropia cruzada

$$Q_n(\mathbf{w}) = - \sum_{i=1}^n [z_i \log z_i - (1 - z_i) \log(1 - z_i)], \quad (13)$$

onde  $z_i = f(\mathbf{x}_i, \mathbf{w})$ .

- Temos que otimizar os pesos  $\mathbf{w}$ , ou seja, encontrar o valor que minimiza  $Q_n(\mathbf{w})$ . Usualmente, temos que obter o gradiente de  $Q_n$ , que vamos indicar por  $\nabla Q_n$ , que aponta para a maior taxa de crescimento de  $Q_n$ . Supondo-se que  $Q_n$  seja uma função contínua suave de  $\mathbf{w}$ , o valor mínimo ocorre no ponto onde o gradiente anula-se.

## O algoritmo backpropagation

- Procedimentos numéricos são usados e há uma literatura extensa sobre o assunto. As técnicas que usam o gradiente, começam fixando-se um valor inicial  $\mathbf{w}^{(0)}$  e os pesos são iterados na forma

$$\mathbf{w}^{(r+1)} = \mathbf{w}^{(r)} - \lambda \nabla Q_n(\mathbf{w}^{(r)}),$$

na qual  $\lambda$  é chamada a **taxa de aprendizado**.

- Esse método é chamado de **gradiente descendente** e usa todo o conjunto de treinamento. É um algoritmo não muito eficiente e, na prática, usa-se o algoritmo **backpropagation** (BP) para calcular o gradiente da SQE em uma RN.

# O algoritmo backpropagation

- Aqui vamos nos basear em (1)–(2) e em Hastie et al. (2017). Podemos usar também (4)–(8), veja Bishop (2006). Vamos chamar de  $\mathbf{w} = (\alpha_0, \dots, \alpha_M, \beta_0, \dots, \beta_K)^\top$  e considerar  $Q_n(\mathbf{w})$  escrita de modo geral como

$$Q_n(\mathbf{w}) = \sum_{k=1}^K \sum_{i=1}^n (z_{ik} - f(\mathbf{x}_i, \mathbf{w}))^2, \quad (14)$$

e seja

$$y_{mi} = h(\boldsymbol{\alpha}_m^\top \mathbf{x}_i), \quad \mathbf{y}_i = (y_{1i}, \dots, y_{Mi})^\top. \quad (15)$$

- Considerando os  $\mathbf{x}_i$  i.i.d, como em (14), escrevemos  $Q_n(\mathbf{w}) = \sum_{i=1}^n Q_i(\mathbf{w})$  e as derivadas

$$\frac{\partial Q_i}{\partial \beta_{km}} = -2(z_{ik} - f(\mathbf{x}_i, \mathbf{w}))g'(\boldsymbol{\beta}_k^\top \mathbf{y}_i)y_{mi}, \quad (16)$$

$$\frac{\partial Q_i}{\partial \alpha_{m\ell}} = -\sum_{k=1}^K 2(z_{ik} - f(\mathbf{x}_i, \mathbf{w}))g'(\boldsymbol{\beta}_k^\top \mathbf{y}_i)\beta_{km}h'(\boldsymbol{\alpha}_m^\top \mathbf{x}_i)x_{i\ell}. \quad (17)$$

# O algoritmo backpropagation

- O gradiente na  $(r + 1)$ -ésima iteração é dado por

$$\beta_{km}^{(r+1)} = \beta_{km}^{(r)} - \lambda_r \sum_{i=1}^n \frac{\partial Q_i}{\partial \beta_{km}^{(r)}}, \quad (18)$$

$$\alpha_{m\ell}^{(r+1)} = \alpha_{m\ell}^{(r)} - \lambda_r \sum_{i=1}^n \frac{\partial Q_i}{\partial \alpha_{m\ell}^{(r)}}, \quad (19)$$

sendo  $\lambda_r$  a taxa de aprendizagem.

- Escrevamos as derivadas (16) e (17) como

$$\frac{\partial Q_i}{\partial \beta_{km}} = \delta_{ki} y_{mi}, \quad (20)$$

$$\frac{\partial Q_i}{\partial \alpha_{m\ell}} = s_{mi} x_{i\ell}, \quad (21)$$

onde  $\delta_{ki}$  e  $s_{mi}$  são os erros do modelo atual na saída e camadas escondidas, respectivamente.

# O algoritmo backpropagation

- De suas definições, esses erros satisfazem

$$s_{mi} = h(\boldsymbol{\alpha}_m^\top \mathbf{x}_i) \sum_{k=1}^K \beta_{km} \delta_{ki}, \quad (22)$$

conhecidas como **equações de backpropagation**, que podem ser usadas para implementar (18) e (19) em duas passadas:

- (a) **uma para a frente**, onde os pesos atuais são fixos e os valores previstos de  $f(\mathbf{x}, \mathbf{w})$  são calculados a partir de (22);
- (b) **uma para trás**, os erros  $\delta_{ki}$  são calculados e propagados para trás via (20) e (21) para obter os erros  $s_{mi}$ .
- Os dois conjuntos de erros são então usados para calcular os gradientes para atualizar (18) e (19) via (20) e (21). Uma **época de treinamento** refere-se a uma passada por todo o conjunto de treinamento.

## Referências

- Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. New York: Springer.
- Hastie, T., Tibshirani, R. and Friedman, J. (2017). *The Elements of Statistical Learning*, 2nd Edition, Springer.
- Hebb, D. O. (1949). *The organization of behavior*. New York: Wiley.
- McCulloch, W. S. and Pitts, W. A. (1943). Logical calculus of the ideas immanent in nervous activity. *Butt. Math. Biophysics*, S, 115–133.
- Pereira, B.B., Rao, C.R. and Oliveira, F. B. (2020). *Statistical Learning Using Neural Networks: A Guide for Statisticians and Data Scientists with Python*. Chapman and Hall.

## Referências

- Rosenblatt, F. (1958). The perceptron: A theory of statistical separability in cognitive systems. Buffalo: Cornell Aeronautical Laboratory, Inc. Report No. VG-1196-G-1.
- Rumelhart, D. E., Hinton, G. E. and Williams, R. J. (1986a). Learning representations by back-propagating errors. *Nature*, **323**, 533–536.
- Rumelhart, D. E., Hinton, G. E. and Williams, R. J. (1986b). Learning internal representations by error propagation. In *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*. Vol. 1: Foundations (eds Rumelhart, D. E. and McClelland, J. L.) 318–362 (MIT, Cambridge, 1986).

# MAE 5905: Introdução à Ciência de Dados

Pedro A. Morettin

Instituto de Matemática e Estatística  
Universidade de São Paulo  
pam@ime.usp.br  
<http://www.ime.usp.br/~pam>

## Aula 16 - Parte 2

15 de junho de 2023

# Sumário

1 Deep Learning

2 Redes Neurais Recorrentes

3 Redes Neurais Long-Short-Term Memory - LSTM

# Deep learning

- Como vimos anteriormente, ML (aprendizado DE máquina) envolve procedimentos segundo os quais um sistema computacional adquire a habilidade de aprender extraindo padrões de dados. Ou seja, tanto na programação clássica, como em ML, nada é conseguido sem dados, que podem tomar a forma de números, vetores, curvas, imagens, palavras etc.
- Vimos que o desempenho de um algoritmo de ML depende da representação desses dados, por meio de funções pré-determinadas, como a logística, tangente hiperbólica etc. A escolha dessa representação é crucial.
- No caso de termos várias camadas intermediárias obtém-se o que é chamado aprendizado profundo (**deep learning** - DL). Nesse caso, o sistema computacional (SC) aprende a partir dos dados (e exemplos) em termos de uma hierarquia de conceitos, cada um definido por meio de sua relação com conceitos mais simples, ou ainda, o SC aprende conceitos complicados a partir de conceitos simples.

# Deep learning

- Um exemplo de modelo DL é o multilayer perceptron, ou seja, uma aplicação que leva os dados de entrada a valores de saída, por meio de composição de funções mais simples.
- Segundo Goodfellow et al. (2016), modelos DL remontam à década de 1940, com os nomes de Cibernética (1940-1960), coneccionismo acoplado a redes neurais (1980-1990) e ressurge como DL em 2006.
- Por sua vez, os tamanhos de conjuntos de dados expandiram-se de small data, da ordem de centenas ou milhares na década de 1900–1980, a dezenas ou centenas de milhares de dados após 1990 (big data). Um exemplo bastante estudado na área é o MNIST (Modified National Institute of Standards and Technology), que consiste de fotos de dígitos escritos a mão.

# Deep learning

- A partir de 2000, conjuntos de dados ainda maiores surgiram, contendo dezenas de milhões de dados, notadamente dados obtidos na Internet. Para analisar esses megadados foi necessário o desenvolvimento de CPU's mais rápidas e as chamadas GPU's (**Graphics Processing Units**). Essas são usadas em celulares, computadores pessoais, estações de trabalho e consoles de jogos e são muito eficientes em computação gráfica e processamento de imagens. Sua estrutura altamente paralela as torna eficientes para algoritmos que processam grandes blocos de dados em paralelo.
- A complexidade de um algoritmo é proporcional ao número de observações, número de preditores, número de camadas e número de épocas de treinamento. Para detalhes sobre esses tópicos, veja Hastie et al. (2017) e Chollet (2018).

# Deep learning

Como dissemos acima, DL tornou-se proeminente nas décadas de 2006–2010. Conforme Chollet (2018), as seguintes aplicações tornaram-se possíveis:

- classificação de imagens ao nível quase humano (NQH);
- reconhecimento de falas ao NQH;
- transcrições de escritas a mão ao NQH;
- aperfeiçoamento de técnicas de ML;
- aperfeiçoamento de conversões textos-para-falas;
- carros autônomos ao NQH;
- aperfeiçoamento de buscas na Internet;
- assistentes digitais, como o Google Now e Amazon Alexa;
- habilidade para responder questões sobre linguagens naturais.

# Deep learning

- Todavia, muitas aplicações levarão muito tempo para serem factíveis e questões relativas ao desenvolvimento, por SC, da inteligência ao nível humano, não devem ser seriamente consideradas no presente estágio de desenvolvimento. Em particular, muitas afirmações feitas nas décadas de 1960–1970 e 1980–1985 sobre **expert systems** e AI em geral, que não se concretizaram, levaram a um decréscimo de investimento em pesquisa na área.
- Em termos de software, há diversas bibliotecas para ML e DL, como **Theano**, **Torch**, **MXNet** e **TensorFlow**.

# Deep learning

- Para analisar redes neurais com várias camadas é possível usar o pacote **keras** do R, que por sua vez usa o pacote **Tensorflow** com capacidades CPU e GPU. Modelos *deep learning* que podem ser analisados incluem **redes neurais recorrentes** (*recurrent neural networks* - RNN), **redes Long Short-Term Memory** (LSTM), **Convolutional Neural Network** (CNN)etc.
- As redes LSTM são apropriadas para captar dependências de longo prazo e implementar previsão de séries temporais.
- As redes CNN representam o estado da arte atualmente, usadas para classificação de imagens, linguagens naturais e outras aplicações.
- Para uma excelente revisão sobre aprendizado profundo veja o artigo de LeCun et al. (2015) e as referências nele contidas, sobre os principais desenvolvimentos na área até 2015. A seguir daremos breves introduções sobre RNN e CNN, com alguns exemplos.

## Redes neurais recorrentes - RNN

- O termo RNN é usado indiscriminadamente para se referir a classes amplas de redes com uma estrutura geral similar. Essas redes exibem um comportamento temporal dinâmico.
- RNN foram baseadas nos trabalhos de David Rumelhart em 1986 e John Hopfield em 1982. LSTM foram inventadas por Hochreiter e Schmidhuber em 1997 e estabeleceram recordes de acurácia em diversas aplicações.
- RNN tem sido um foco importante de pesquisa desde a década de 1990. Elas são planejadas para aprender padrões sequenciais ou variando no tempo.
- RNN têm sido aplicada a uma variedade grande de problemas. No final da década de 1980 foram introduzidas RNN parciais por vários pesquisadores, como Rumelhart, Hinton e Williams (veja Rumelhart, 1986) para aprender sequências de caracteres. Outras aplicações incluem problemas em sistemas dinâmicos com sequências temporais de eventos, estimação da potência de turbinas, previsões financeiras, sínteses musicais, previsões de cargas elétricas, previsão de vazões de rios etc.

## Redes neurais recorrentes - RNN

- A arquitetura de RNN pode ser:
  - (a) **completamente interconectada**: todos os estados são alcançados por qualquer estado, inclusive cada estado de si próprio;
  - (b) **parcialmente conectada**: nem todos os estados são alcançados da partir de um estado qualquer. Essas redes são usadas para aprender sequência de caracteres.
- Pesquisadores desenvolveram uma variedade de esquemas pelos quais métodos gradiente, e em especial, aprendizado por backpropagation, podem ser estendidos a RNN. Werbos introduziu um procedimento que aproxima a evolução de uma RNN como uma sequência de redes estáticas usando métodos gradiente (*backpropagation through time*). Outras abordagens podem ser encontradas em Lapedes e Farber(1986), Pineda (1987), Almeida (1987), Sato (1990). As várias tentativas para estender backpropagation para RNN estão sumarizadas em Pearlmutter (1995).

## Redes neurais recorrentes - RNN

- Do ponto de vista de séries temporais, RNN podem ser vistas como modelos não lineares de espaços de estados. Veja Morettin e Toloi (2020) para detalhes sobre esses modelos.
- Uma RNN processa um elemento por vez de uma sequência de entrada  $X_t$ , mantendo sua unidades ocultas num **vetor de estados**  $S_t$ , que contém informação sobre a história passada da sequência (LeCun et al. 2015). Denotando por  $Y_t$  a sequência processada no estado  $t$ , uma rede neural recorrente processa sequências de entradas iterativamente,

$$Y_t = f(Y_{t-1}, X_t),$$

e as previsões são feitas segundo

$$\hat{Y}_{t+h|t} = g(h_t),$$

com  $f$  e  $g$  denotando funções a serem definidas.

- Todas as redes neurais recorrentes têm a forma de uma rede neural que se repete como na Figura 1: A, B e W são parâmetros (matrizes) invariantes no tempo. O algoritmo BP pode ser aplicado à rede desdobrada.

## Redes neurais recorrentes - RNN

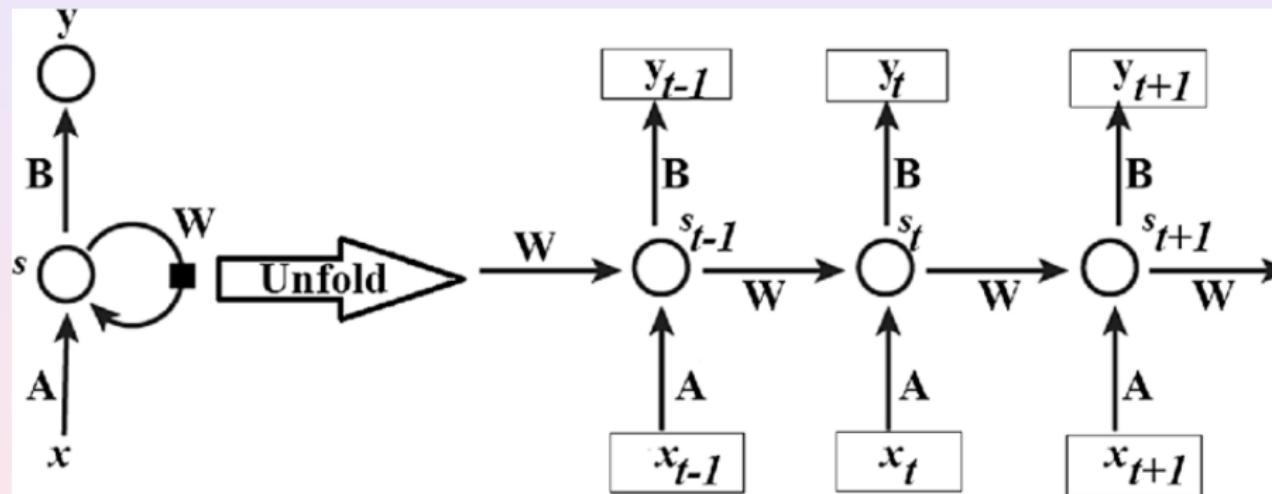


Figura 1: RNN e seu desdobramento no tempo.

## Redes neurais recorrentes - RNN

- Redes neurais recorrentes são difíceis de implementar pois sofrem do chamado **problema do gradiente evanescente** (*vanishing gradient problem*).
- Uma solução foi proposta por Hochreiter e Schmidhuber (1997) por meio de uma variante das redes neurais recorrentes, chamada **rede com memórias de curto e longo prazos** (*Long-Short-Term Memory - LSTM*), capazes de “aprender” dependências de longo prazo.
- No caso de séries temporais, há modelos para descrever processos com memória curta (*short memory*), como os modelos *ARIMA* (autorregressivos, integrados e de média móveis), e modelos para descrever memória longa (*long memory*), como os modelos *ARFIMA* (autorregressivos, integrados fracionários e de médias móveis). Veja, por exemplo, Morettin e Toloi (2018).
- Uma rede *LSTM* modela simultaneamente as memórias de curto e longo prazos. Discutimos brevemente essas redes na seção seguinte.

# LSTM

- Cerca de 2007, LSTMs revolucionaram o reconhecimento de fala, superando modelos tradicionais em aplicações nessa área. Em 2009, uma rede **Connectionist Temporal Classification (CTC)-trained LSTM** foi a primeira RNN a vencer competições em reconhecimento de padrões.
- LSTMs também melhoraram o reconhecimento de grandes vocabulários de falas e sínteses de textos para falas e foi usada no Google Android. Em 2015, o reconhecimento de falas do Google obteve um salto de desempenho da ordem de 49% por meio de uma rede **CTC-trained LSTM**.
- Redes LSTM quebraram recordes para melhorar traduções de textos, modelagem de linguagens e processamento de múltiplas línguas. LSTM combinada com **convolutional neural networks (CNNs)** melhoraram a captação automática de imagens.

# LSTM

- Uma rede LSTM consiste de blocos de memória, chamadas **células** (*cells*), conectadas por meio de camadas (*layers*).
- A informação nas células está contida no estado  $C_t$  e no estado escondido  $h_t$  e é regulada por mecanismos chamados **portas** (*gates*), por meio de funções de ativação *sigmoid* e *tanh*.
- A função sigmoide tem como saídas números entre 0 e 1, com 0 indicando *nada passa* e 1 indicando *tudo passa*. A rede LSTM tem, portanto, a habilidade de adicionar ou desprezar informação do estado da célula (condicionalmente).
- Em geral, as portas têm como entradas os estados escondidos do instante anterior,  $h_{t-1}$  e da entrada atual  $x_t$  e as multiplica por pesos matriciais,  $\mathbf{W}$ , e um viés (*bias*)  $b$  é adicionado ao produto.

# LSTM

Há três portas principais:

- (i) **Forget gate**: esta porta determina qual informação será desprezada do estado da célula. A saída será 0, significando **apagar** e 1, implicando **lembrar todos**; formalmente,

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f).$$

- (ii) **Input gate**: nesse passo, a função de ativação *tanh* criará um vetor de candidatos potenciais como segue:

$$\hat{C}_t = \tanh(W_c[h_{t-1}, x_t] + b_c).$$

A camada sigmoide cria um filtro atual como segue:

$$U_t = \sigma(W_u[h_{t-1}, x_t] + b_u).$$

A seguir o estado anterior  $C_{t-1}$  é atualizado como

$$C_t = f_t C_{t-1} + U_t \hat{C}_t.$$

# LSTM

- (iii) **Output gate:** nesse passo, a camada sigmoide filtra o estado que vai para a saída:

$$O_t = \sigma(W_0[h_{t-1}, x_t] + b_0).$$

- (iv) O estado  $C_t$  é então passado por meio da função  $\tanh$  para escalar os valores para o intervalo  $[-1, 1]$ . Finalmente, o estado escalonado é multiplicado pela saída filtrada para se obter o estado escondido  $h_t$  a ser passado para a próxima célula:  $h_t = O_t \times \tanh(C_t)$ .

A arquitetura de uma rede LSTM está mostrada na Figura 2.

# LSTM

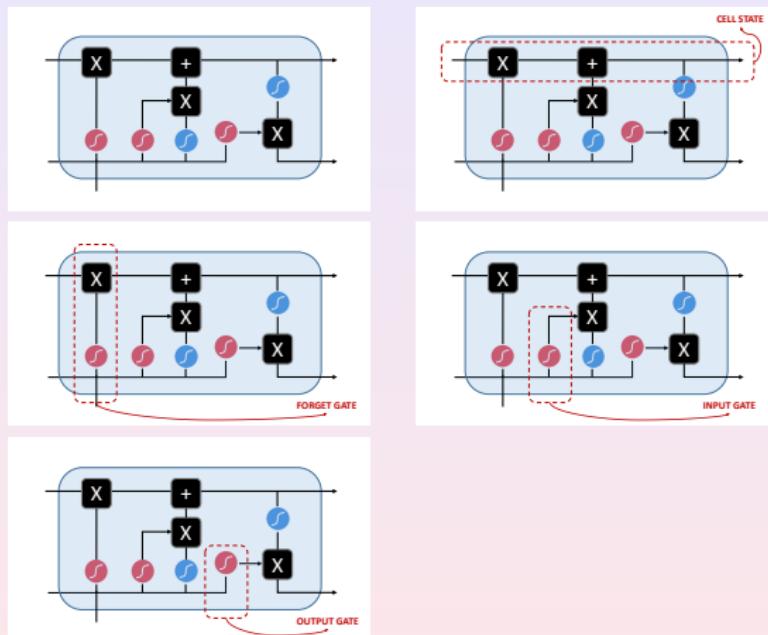


Figura 2: Arquitetura da célula de uma rede LSTM.

## LSTM - Exemplo 1

- Consideremos os preços diárias das ações da Petrobrás entre 31/08/1998 e 29/09/2010, totalizando  $n = 2990$  observações, constantes do arquivo acoes.
- O gráfico da série está indicado na Figura 3, mostrando o seu caráter não estacionário.
- Como a rede LSTM funciona melhor para séries estacionárias ou mais próximas possíveis da estacionariedade, tomemos a série de primeiras diferenças, definida como  $Y_t = X_t - X_{t-1}$ , com  $X_t$  denotando a série original. A série de primeiras diferenças também está apresentada na Figura 3.
- Depois de obtidas as previsões deve-se fazer a transformação inversa para obter as previsões com a série original.
- Nosso intuito é fazer previsões para os dados de um conjunto de validação, com 897 observações, ajustando uma rede LSTM aos dados de um conjunto de treinamento, consistindo de 2093 observações.

## LSTM - Exemplo 1

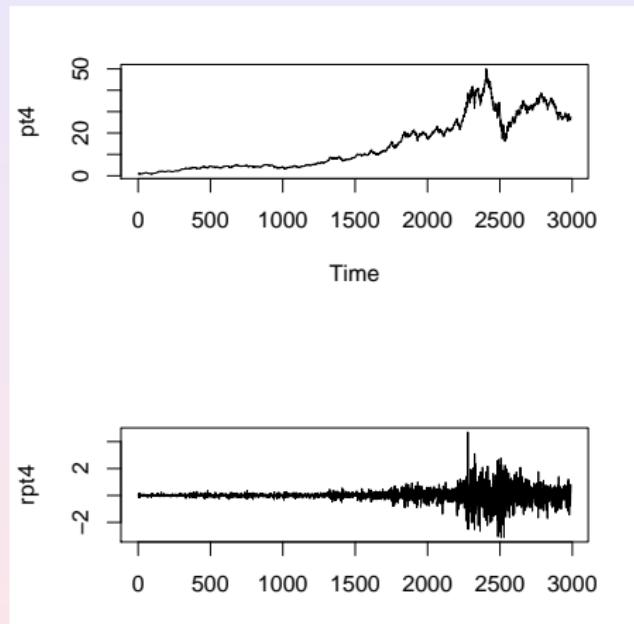


Figura 3: Série de preços das ações da Petrobrás e respectivas diferenças.

## LSTM - Exemplo 1

- Como as redes LSTM supõem dados na forma de aprendizado supervisionado, ou seja, com uma variável resposta  $Y$  e uma variável preditora  $X$ , tomamos valores defasados da série de forma que os valores obtidos até o instante  $t - k$  serão considerados como variáveis preditoras e o valor no instante  $t$  será a variável resposta. Neste exemplo, tomamos  $k=1$ .
- Para esse efeito, consideramos os comandos

```
lag_transform <- function(x, k=1){  
  lagged = c(rep(NA,k),x[1:(length(x)-k)])  
  DF = as.data.frame(cbind(lagged,x))  
  colnames(DF) <- c(paste0('x-',k),'x')  
  DF[is.na(DF)] <- 0  
  return(DF)  
}  
  
supervised = lag_transform(Petrobras4, 1)  
head(supervised)
```

## LSTM - Exemplo 1

- obtendo, para as 6 primeiras observações, a variável preditora na coluna  $x-1$  e a variável resposta, na coluna  $x$ .

|   | $x-1$ | $x$   |
|---|-------|-------|
| 1 | 0.00  | 0.08  |
| 2 | 0.08  | 0.07  |
| 3 | 0.07  | -0.06 |
| 4 | -0.06 | -0.04 |
| 5 | -0.04 | -0.04 |
| 6 | -0.04 | -0.03 |

- Para a divisão entre dados treinamento (70%) e de validação (30%), podemos utilizar os comandos

```
N <- nrow(supervised)
n <- round(N*0.7,digits=0)
train <- supervised[1:n,]
test <- supervised[(n+1):N,]
```

## LSTM - Exemplo 1

- Em seguida, normalizamos os dados de entrada para que pertençam ao intervalo de variação da função de ativação, que é a sigmoide, com variação em [-1,1].
- Os valores mínimo e máximo do conjunto de treinamento são usados para normalizar os conjuntos de treinamento e de validação além dos valores preditos, por meio dos comandos

```
scale_data <- function(train, test, feature_range=c(0,1)){  
  x<-train  
  fr_min<-feature_range[1]  
  fr_max<-feature_range[2]  
  std_train<-((x-min(x))/(max(x)-min(x)))  
  std_test<-(test-min(x))/(max(x)-min(x))  
  scaled_train<-std_train*(fr_max-fr_min)+fr_min  
  scaled_test<-std_test*(fr_max-fr_min)+fr_min  
  return(list(scaled_train=as.vector(scaled_train),  
             scaled_test=as.vector(scaled_test),  
             scaler=c(min=min(x),max=max(x))))  
}
```

## LSTM - Exemplo 1

- continuando:

```
Scaled <- scale_data(train,test, c(-1,1))  
y_train <- Scaled$scaled_train[,2]  
x_train <- Scaled$scaled_train[,1]  
y_test <- Scaled$scaled_test[,2]  
x_test <- Scaled$scaled_test[,1]
```

- Para reverter os valores previstos à escala original, consideramos

```
invert_scaling <- function(scaled, scaler, feature_range=c(0,1)){  
  min=scaler[1]  
  max=scaler[2]  
  t=length(scaled)  
  mins=feature_range[1]  
  maxs=feature_range[2]  
  inverted_dfs=numeric(t)  
  for(i in 1:t){  
    X=(scaled[i]-mins)/(maxs-mins)  
    rawValues=X*(max-min)+min  
    inverted_dfs[i]<-rawValues  
  }  
  return(inverted_dfs)
```

## LSTM - Exemplo 1

- A partir deste ponto iniciamos a modelagem. Com essa finalidade, precisamos fornecer o lote de entrada na forma de um vetor tridimensional, [samples, timesteps, features] a partir do estado atual [samples, features], em que samples é o número de observações em cada lote (tamanho do lote), timesteps é o número de passos para uma dada observação (para este exemplo, timesteps=1) e features=1, para o caso univariado como no exemplo.
- O tamanho do lote deve ser função dos tamanhos das amostras de treinamento e de validação. Usualmente esse valor é 1. Também devemos especificar stateful = TRUE de modo que após processar um lote de amostras os estados internos sejam reutilizados para as amostras do lote seguinte. Os comandos correspondentes são

```
dim(x_train) <- c(length(x_train), 1, 1)
X_shape2 <- dim(x_train)[2]
X_shape3 <- dim(x_train)[3]
batch_size=1
units=1
model <- keras_model_sequential()
model %>% layer_lstm(units,
  batch_input_shape = c(batch_size, X_shape2, X_shape3),
  stateful=TRUE) %>% layer_dense(units=1)
```

## LSTM - Exemplo 1

- O modelo pode, então ser compilado, com a especificação do erro quadrático médio como função perda.
- O algoritmo de otimização é o *Adaptive Monument Estimation (ADAM)*. Usamos a acurácia como métrica para avaliar o desempenho do modelo.

```
model %>% compile(  
  loss='mean_squared_error', optimizer=optimizer_adam(lr=0.02, decay=1e-6),  
  metrics=c('accuracy'))
```

Por meio do comando `summary(model)` obtemos:

Model: "sequential"

| Layer (type)            | Output Shape | Param # |
|-------------------------|--------------|---------|
| lstm (LSTM)             | (1, 1)       | 12      |
| dense (Dense)           | (1, 1)       | 2       |
| Total params: 14        |              |         |
| Trainable params: 14    |              |         |
| Non-trainable params: 0 |              |         |

## LSTM - Exemplo 1

- Para ajustar o modelo, a rede LSTM exige o comando `shuffle=FALSE` cuja finalidade é evitar o embaralhamento do conjunto de treinamento e manter a dependência entre  $x_i$  e  $x_{i+t}$ .
- Além disso o algoritmo requer a redefinição do estado da rede após cada iteração (época).
- Os comandos para o ajuste do modelo são

```
Epochs=50
for(i in 1:Epochs){
  model %>% fit(x_train, y_train, epochs=1, batch_size=batch_size,
  verbose=1,shuffle=FALSE)
  model %>% reset_states()}
```

são geradas 50 épocas (iterações) da rede neural e em cada época é possível visualizar o valor da função perda e a acurácia.

# LSTM - Exemplo 1

- Para fazer previsões usamos a função predict() e em seguida invertemos a escala e as diferenças para retornar à série original.

```

L=length(x_test)
scaler=Scaled$scaler
predictions=numeric(L)
for(i in 1:L){
  X=x_test[i]
  dim(X)=c(1,1,1)
  yhat=model %>% predict(X, batch_size=batch_size)
  yhat=invert_scaling(yhat, scaler, c(-1,1))
  yhat
  yhat=yhat+Petrobras4[(n+i)]
  predictions[i] <- yhat}

```

- As previsões para as 897 observações do conjunto teste podem ser obtidas por meio da função prediction(). As 6 primeiras são apresentadas abaixo:

```
[1] -0.28833461  0.35166539 -0.16833461 -0.26833461  0.01166539
-0.03833461
```

## LSTM - Exemplo 1

- As observações do conjunto de validação e as correspondentes previsões estão representadas na Figura 4.
- Na Figura 5 colocamos a série para o conjunto de treinamento (em azul), a série no conjunto de validação (em vermelho) e a série de previsões (em verde).
- A raiz quadrada do erro quadrático médio, calculado da maneira habitual, é  $RMSE = 0,02835$ .

## LSTM - Exemplo 1

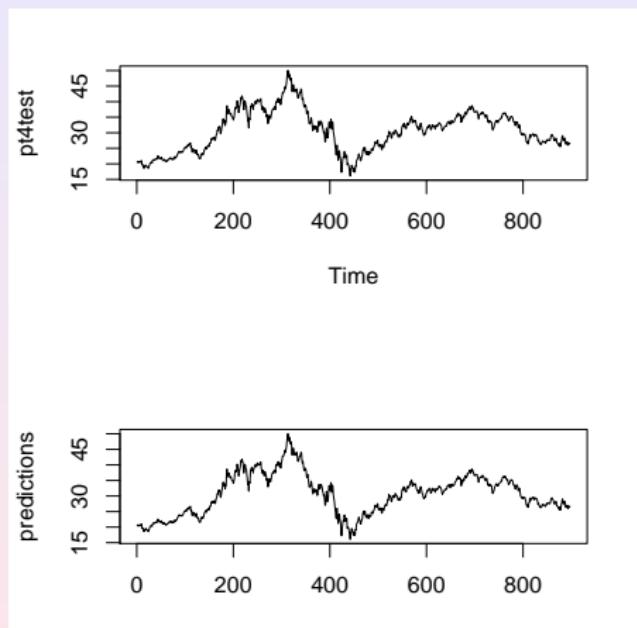


Figura 4: Série original e série de previsões de preços das ações da Petrobrás, no conjunto de validação.

## LSTM - Exemplo 1

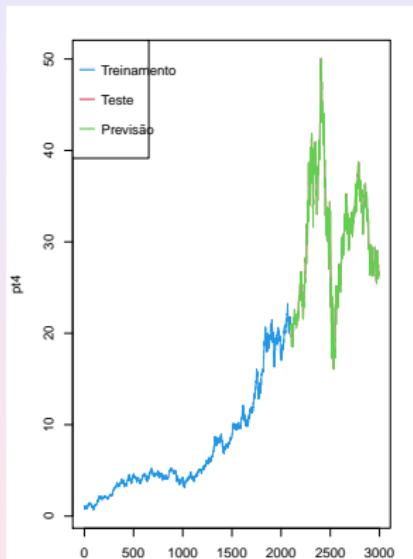


Figura 5: Série original de preços das ações da Petrobrás para o conjunto de treinamento (azul), para o conjunto de validação (vermelho) e série de previsões (verde).

## LSTM - Exemplo 2

- Consideraremos os dados do arquivo covid2 com 526 observações de casos e óbitos por COVID-19 no Brasil entre 19/03/2020 e 25/08/2021.
- O gráfico da série está na Figura 6.
- Separemos  $n = 369$  observações para o conjunto de treinamento e  $m = 157$  observações para o conjunto de validação. O objetivo é prever o número de casos no conjunto de validação.
- Após uma análise por meio de uma rede LSTM similar àquela do Exemplo 1, o comando `summary()` produz o seguinte resultado:

## LSTM - Exemplo 2

```
summary(model)
Model: "sequential"

-----
Layer (type)          Output Shape         Param #
=====
lstm (LSTM)           (1, 1)                 12
-----
dense (Dense)         (1, 1)                 2
-----
Total params: 14
Trainable params: 14
Non-trainable params: 0
-----
```

## LSTM - Exemplo 2

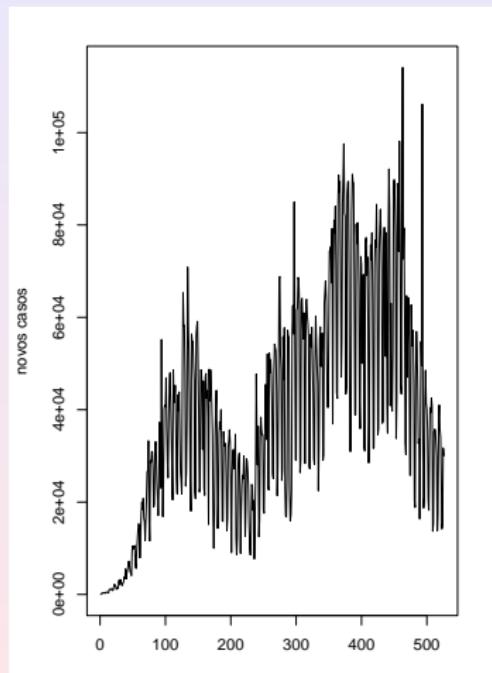


Figura 6: Série diária de novos infectados pelo vírus Covid-19 no Brasil.

## LSTM - Exemplo 2

A raiz quadrada do erro quadrático médio é  $RMSE = 20793,87$  e as previsões podem ser vistas na Figura 7.

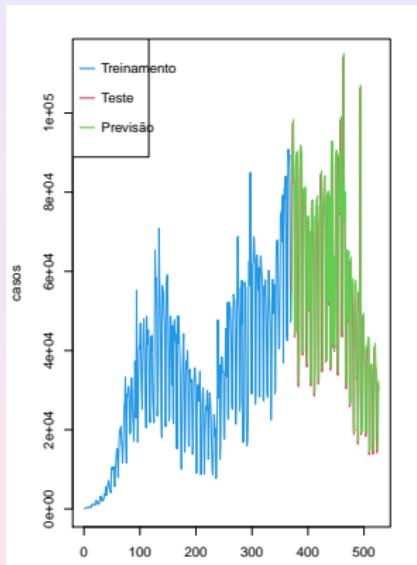


Figura 7: Série original de novos infectados por Covid-19 no Brasil para o conjunto de treinamento (azul), para o conjunto de validação (vermelho) e série de previsões (verde).

## Referências

- Chollet, F. (2018). *Deep Learning with R*. Manning.
- Goodfellow, I., Bengio, Y. and Courville, A. (2016). *Deep Learning*. The MIT Press.
- Hastie, T., Tibshirani, R. and Friedman, J. (2017). *The Elements of Statistical Learning*, 2nd ed. New York: Springer.
- LeCun, Y., Bengio, Y. and Hinton, G. (2015). Deep learning. *Nature*, **521**, 436–444.
- Morettin, P. A. e Singer, J. M. (2023). *Estatística e Ciência de Dados*. Segunda edição. LTC: Rio de Janeiro.

# MAE 5905: Introdução à Ciência de Dados

Pedro A. Morettin

Instituto de Matemática e Estatística  
Universidade de São Paulo  
pam@ime.usp.br  
<http://www.ime.usp.br/~pam>

## Aula 16 - Parte 3

20 de junho de 2023

# Sumário

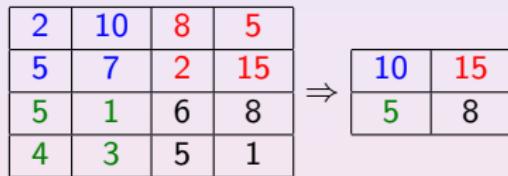
1 Redes neurais convolucionais - CNN

2 Redes generativas adversárias - GAN

# CNN

- As redes neurais convolucionais (*convolutional neural networks*, CNN) foram introduzidas por LeCun et al. (1998) e são muito eficientes para resolver problemas de classificação de imagens.
- De modo geral, as CNN são desenhadas para processar dados que surgem na forma de matrizes (*arrays*), como sequências (séries temporais e linguagem) unidimensionais, sinais de audio bidimensionais e audio e imagens tridimensionais. Como exemplo, podemos ter uma imagem colorida composta de três *arrays* bidimensionais, contendo intensidades de pixel nos três canais de cores (RGB) (LeCun et al., 2015).
- Nas CNN, para extrair padrões (*features*) dos dados, há quatro ideias básicas:
  - uso de muitas camadas;
  - camadas de convolução (*convolution layers*, CL);
  - camadas de agrupamento (*pooling layers*, PL);
  - camadas totalmente conectadas.

- A CL é responsável por extrair os padrões locais (*feature maps*) da camada precedente. Esta é feita por meio de pesos (*filter banks*) de tamanhos reduzidos. Diferentes bancos de filtros são usados para os diversos padrões da imagem, como arestas, arranjos de arestas, partes de objetos familiares, cores etc. O resultado dessa soma ponderada local é passada por meio de uma não linearidade (ReLU).
- A PL, usada após uma camada convolucional, destina-se a reduzir a dimensão dos dados de entrada e juntar características locais numa só. Pode-se usar médias ou escolher o maior valor encontrado em subregiões. Este segundo procedimento é o mais utilizado e chamado *maxpooling*. Na Figura 1 temos um exemplo com uma imagem  $4 \times 4$  e um maxpooling com filtro  $2 \times 2$ . Essa técnica reduz a quantidade de dados para a camada seguinte, reduzindo também o custo de processamento e memória.



**Figura 1:** Exemplo de *maxpooling* no caso de uma imagem  $4 \times 4$ .

# CNN

- As camadas totalmente conectadas situam-se no final da rede e os padrões extraídos nas camadas de convolução anteriores são utilizadas para a classificação final da rede neural.
- As razões para essa arquitetura são (LeCun et al, 2015):
  - (i) em *arrays*, como imagens, grupos locais de valores são correlacionados, formando padrões (*motifs*) facilmente detectados;
  - (ii) as estatísticas locais dessas *arrays* são invariantes em relação à localização, donde a ideia de que os mesmos pesos são compartilhados por unidades em diferentes localizações.
- Os cálculos com CNN envolvem contrações em escalas múltiplas, linearização de simetrias hierárquicas e separação esparsa. Em muitas aplicações o número de amostras cresce linearmente com o número de dimensões.
- Como todo algoritmo de aprendizagem, uma CNN é baseada em alguma suposição de suavidade (regularidade) do classificador, digamos,  $f(\mathbf{x})$ , sendo  $\mathbf{x}$  o vetor de dados, e a natureza dessa regularidade é o problema matemático mais importante.
- A ideia é reduzir a dimensão de  $\mathbf{x}$  e isso pode ser feito definindo-se uma nova variável  $\phi(\mathbf{x})$ , em que  $\phi$  é um operador contração, que reduz a variabilidade de  $\mathbf{x}$ , aliada à separação de valores distintos de  $f(\mathbf{x})$ . Os aspectos matemáticos de uma CNN estão descritas em Mallat (2026) e Kohler et al. (2022).

# CNN - Exemplo 1

- Vejamos um exemplo de convolução com uma série temporal fictícia com  $n = 10$  observações como entrada:

|     |     |      |     |     |      |      |     |     |     |
|-----|-----|------|-----|-----|------|------|-----|-----|-----|
| 1,2 | 0,9 | -0,8 | 0,7 | 1,5 | -1,3 | -1,0 | 0,7 | 1,3 | 1,4 |
|-----|-----|------|-----|-----|------|------|-----|-----|-----|

Consideremos um filtro com coeficientes:

|   |   |   |
|---|---|---|
| 1 | 2 | 1 |
|---|---|---|

- A convolução dos três primeiros valores da série com os pesos do filtro resulta  $(1,2) \times 1 + (0,9) \times 2 + (-0,8) \times 1 = 2,2$ . Deslocando-se uma unidade de tempo e efetuando o produto dos valores seguintes pelos coeficientes do filtro obtemos o valor 0. Continuando, obtemos a série de saída

|     |   |     |     |      |      |     |     |
|-----|---|-----|-----|------|------|-----|-----|
| 2,2 | 0 | 2,1 | 2,4 | -2,1 | -2,6 | 1,7 | 4,7 |
|-----|---|-----|-----|------|------|-----|-----|

# CNN - Exemplo 1

- Para que tenhamos convolução e *maxpooling*, temos que adicionar dois zeros no começo e final da série (*padding*):

|   |   |     |     |      |     |     |      |      |     |     |     |   |   |
|---|---|-----|-----|------|-----|-----|------|------|-----|-----|-----|---|---|
| 0 | 0 | 1,2 | 0,9 | -0,8 | 0,7 | 1,5 | -1,3 | -1,0 | 0,7 | 1,3 | 1,4 | 0 | 0 |
|---|---|-----|-----|------|-----|-----|------|------|-----|-----|-----|---|---|

- A série convolvida e a saída após tomar o máximo de cada três observações estão mostradas a seguir:

|     |     |     |   |     |     |      |      |     |     |     |     |
|-----|-----|-----|---|-----|-----|------|------|-----|-----|-----|-----|
| 1,2 | 3,3 | 2,2 | 0 | 2,1 | 2,4 | -2,1 | -2,6 | 1,7 | 4,7 | 4,1 | 1,4 |
|-----|-----|-----|---|-----|-----|------|------|-----|-----|-----|-----|

|     |     |     |     |
|-----|-----|-----|-----|
| 3,3 | 2,4 | 1,7 | 4,7 |
|-----|-----|-----|-----|

- A Figura 2 ilustra uma CNN com série temporal como entrada. Se a entrada for outra *array*, como uma imagem, o esquema é o mesmo, obtendo-se não mais sequências unidimensionais, mas matrizes, como na Figura 1.

## CNN - Exemplo 1

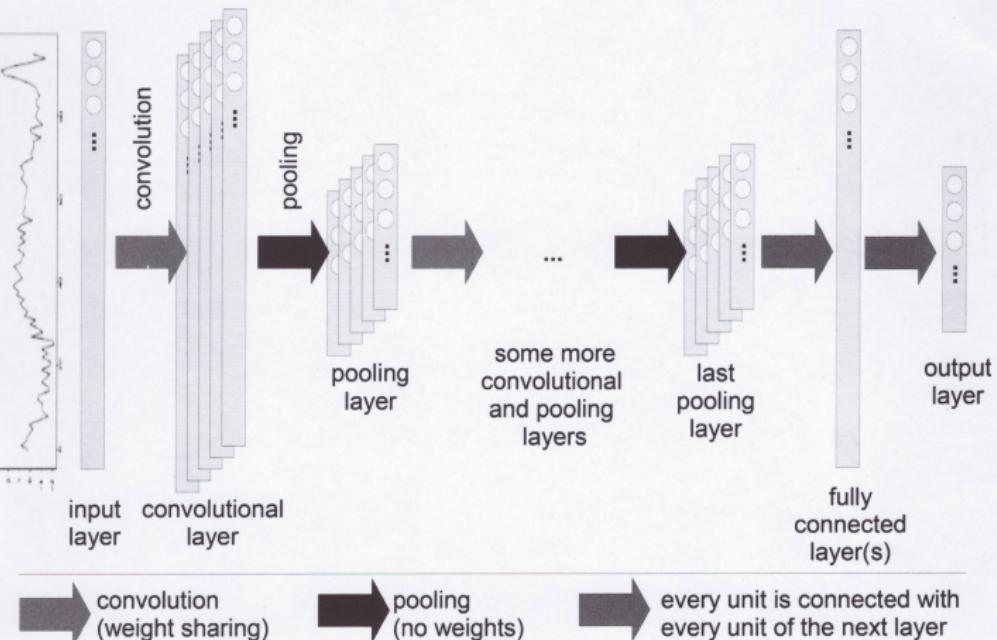


Figure 2: Rede neural convolucional.

## CNN - Exemplo 2

- Vamos usar os dados de fotos de dígitos escritos a mão, **Mnist** (Modified NIST(National Institute of Standards and Technology)).
- Esses dados contém 60.000 imagens de treinamento e 10.000 imagens de teste, dos quais a metade de cada conjunto foi retirada conjunto de treinamento do NIST, as outras duas metades foram retiradas do conjunto teste do NIST.
- Esses dados foram usados com diversos tipos de classificadores, dentre os quais classificadores lineares, KNN, SVM, florestas aleatórias e diversos tipos de redes neurais, incluindo as CNN. Usaremos um código constante do site

[https://rpubs.com/juanhklopper/example\\_of\\_a\\_CNN](https://rpubs.com/juanhklopper/example_of_a_CNN)

# CNN - Exemplo 2

Comentaremos alguns comandos:

- A primeira imagem é um 5:

```
> y_train[1,] # a primeira imagem \'e um 5
```

```
[1] 0 0 0 0 0 1 0 0 0 0
```

- Criando o modelo:

```
model <- keras_model_sequential() %>%
  layer_conv_2d(filters = 16,
                 kernel_size = c(3,3),
                 activation = 'relu',
                 input_shape = input_shape) %>%
  layer_max_pooling_2d(pool_size = c(2, 2)) %>%
  layer_dropout(rate = 0.25) %>%
  layer_flatten() %>%
  layer_dense(units = 10,
              activation = 'relu') %>%
  layer_dropout(rate = 0.5) %>%
  layer_dense(units = num_classes,
              activation = 'softmax')
```

## CNN - Exemplo 2

- O comando `summary()` mostra que foram aprendidos 27.320 parâmetros.

```
> model %>% summary(),
```

| Layer (type)                 | Output Shape       | Param # |
|------------------------------|--------------------|---------|
| <hr/>                        |                    |         |
| conv2d (Conv2D)              | (None, 26, 26, 16) | 160     |
| max_pooling2d (MaxPooling2D) | (None, 13, 13, 16) | 0       |
| dropout_1 (Dropout)          | (None, 13, 13, 16) | 0       |
| flatten (Flatten)            | (None, 2704)       | 0       |
| dense_1 (Dense)              | (None, 10)         | 27050   |
| dropout (Dropout)            | (None, 10)         | 0       |
| dense (Dense)                | (None, 10)         | 110     |
| <hr/>                        |                    |         |

Total params: 27,320

Trainable params: 27,320

Non-trainable params: 0

---

## CNN - Exemplo 2

- O modelo é compilado com a função perda entropia cruzada e métrica acurácia:

```
model %>% compile(  
  loss = loss_categorical_crossentropy,  
  optimizer = optimizer_adadelta(),  
  metrics = c('accuracy')  
)
```

- Número de mini-batches e número de épocas:

```
batch_size <- 128  
epochs <- 50
```

- Treinando o modelo com 50 iterações (épocas):

```
model %>% fit(  
  x_train, y_train,  
  batch_size = batch_size,  
  epochs = epochs,  
  validation_split = 0.2  
)
```

## CNN - Exemplo 2

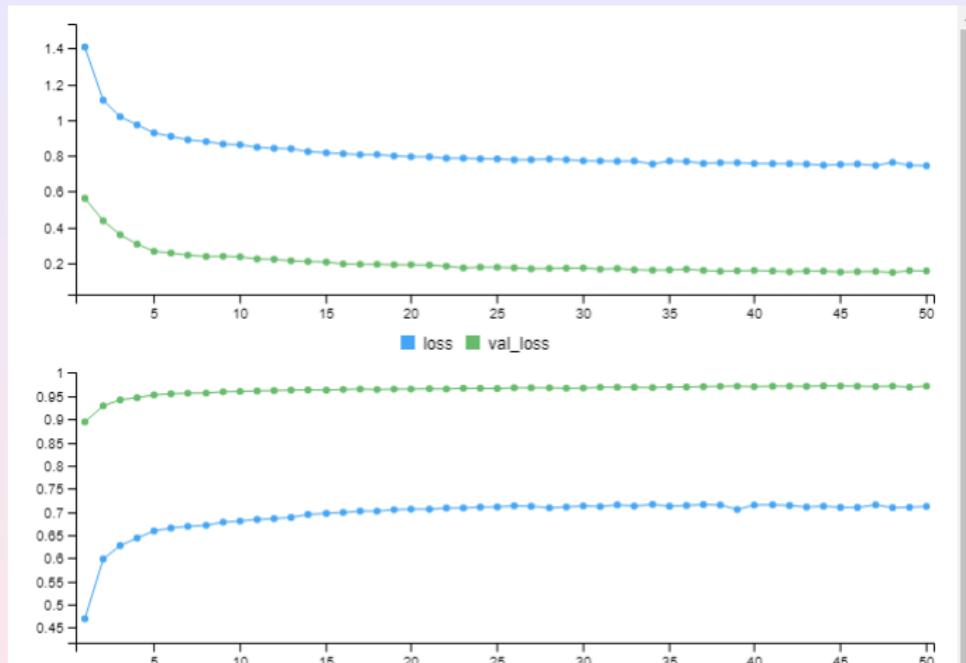
- Avaliando a perda e acurácia no conjunto teste:

```
> score <- model \%>\% evaluate(x_test,  
+                                     y_test)  
313/313  - 1s 2ms/step - loss: 0.1569 - accuracy: 0.9723
```

- Ou seja, a perda (dada pela entropia cruzada) no conjunto teste foi 0,1569 e a acurácia no conjunto teste 0,9723.
- O gráfico da Figura 3 mostra a evolução das perdas (medidas pela entropia cruzada) e da acurácia do procedimento.
- Um código em Python para o mesmo conjunto de dados pode ser encontrado em

[https://keras.io/examples/vision/mnist\\_convnet/](https://keras.io/examples/vision/mnist_convnet/)

## CNN - Exemplo 2



**Figura 3:** Perdas (entropia, painel superior) e acurácia(painel inferior) para o Exemplo 2.

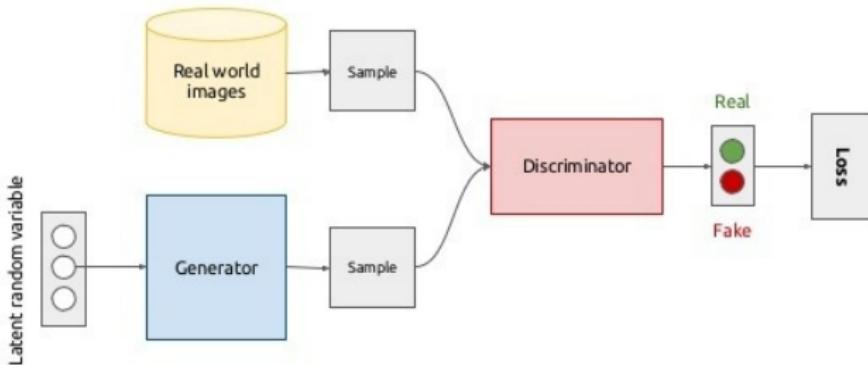
# Redes generativas adversárias

- As redes generativas adversárias (*generative adversarial networks*, GAN) foram introduzidas por Goodfellow et al. (2014). Elas consistem de duas redes neurais treinadas em oposição uma à outra: o **gerador** (generator) G e o **discriminador** (discriminator) D.
- Considere uma matriz de dados  $\mathbf{X}$ , e suponha que temos exemplos, considerados amostras i.i.d. de  $\mathbf{X}$ , com densidade de probabilidade  $p(\mathbf{x})$ , desconhecida.
- O gerador G tem como entrada um vetor de ruídos aleatórios  $\mathbf{z}$  e saída uma imagem  $X_{\text{falso}} = G(\mathbf{z})$ , com a mesma distribuição  $p(\mathbf{x})$  dos dados reais.
- Por sua vez, o discriminador D depara-se com um exemplo real,  $\mathbf{X} \sim p(\mathbf{x})$  ou um exemplo falso,  $X_{\text{falso}}$  gerado por G.
- Quem é apresentado é decidido jogando-se uma moeda honesta (a probabilidade de ser falso é  $1/2$ ). A moeda é lançada independentemente em cada rodada do jogo.
- O discriminador usa uma função  $D(\mathbf{x}) = P(\mathbf{x} \text{ real})$  (o classificador), tal que se  $D(\mathbf{x}) > 0,5$  então  $\mathbf{x}$  é real; se  $D(\mathbf{x}) < 0,5$ , então  $\mathbf{x}$  é falso.

# Redes gerativas adversárias

- Essa é a **regra de classificação**. D gostaria que  $D(x) \approx 1$  quando  $x$  for real e  $D(x) \approx 0$  quando falso.
- Na Figura 4 temos o esquema de uma rede neural GAN.

## Generative adversarial networks (conceptual)



**Figura 4:** Rede neural geradora adversária.

# Redes generativas adversárias

- Para gerar  $\mathbf{X}$  a partir de  $\mathbf{Z}$  usamos o Método Monte Carlo. De modo geral, a função de distribuição acumulada (f.d.a.) de um vetor  $\mathbf{X} = (X_1, \dots, X_d)$  pode ser escrita como

$$F_{X_1, \dots, X_d}(x_1, \dots, x_d) = F_{X_1}(x_1)F_{X_2|X_1}(x_2|x_1) \cdots F_{X_d|X_1, \dots, X_{d-1}}(x_d|x_1, \dots, x_{d-1}).$$

- A seguir, geramos  $d$  variáveis aleatórias i.i.d.  $\mathcal{U}(0, 1)$ , digamos  $U_1, \dots, U_d$  e geramos o vetor  $\mathbf{X}$  por meio de:

$$\begin{aligned} x_1 &= F_{X_1}^{-1}(u_1), \\ x_2 &= F_{X_2|X_1}^{-1}(u_2) \\ &\vdots \\ x_d &= F_{X_d|X_1, \dots, X_{d-1}}^{-1}(u_d). \end{aligned}$$

- Em nosso caso, sabemos que existe uma função  $G$  que transforma ruídos  $\mathbf{Z}$  num vetor  $\mathbf{X}$  com a densidade  $f(\mathbf{x})$ . Mas nesse caso, não conhecemos a distribuição de  $\mathbf{X}$  e será difícil obter a f.d.a.

# Redes generativas adversárias

- Como mencionamos acima, a solução é imaginar que exista essa  $G$  na forma de uma rede neural e o gerador  $G(\mathbf{z})$  terá parâmetros  $\theta^{(G)}$  (os pesos da rede neural). Teremos que aprender os parâmetros dessa rede de forma aproximada.
- A função perda do discriminador é dada por

$$J^{(D)}(\theta) = -\frac{1}{2} E_{\mathbf{p} \text{ real}}[\log(D(\mathbf{X}))] - \frac{1}{2} E_{\mathbf{z}}[\log((1 - D(G(\mathbf{z})))]. \quad (1)$$

- O tipo de jogo a usar é o de soma zero: a perda de um jogador é o ganho do outro e a soma das perdas dos dois jogadores é zero. Neste caso, a função perda do gerador é dada por

$$J^{(G)}(\theta^{(G)}, \theta^{(D)}) = -J^{(D)}(\theta^{(G)}, \theta^{(D)}). \quad (2)$$

- A solução de um jogo de soma zero é chamada **solução minimax**. O gerador  $G$  quer determinar seus pesos  $\theta^{(G)}$  que minimizem sua perda, dada por

$$J^{(G)}(\theta^{(G)}, \theta^{(D)}) = \frac{1}{2} E_{\mathbf{p} \text{ dados}}[\log(D(\mathbf{X}))] + \frac{1}{2} E_{\mathbf{z}}[\log((1 - D(G(\mathbf{z})))]. \quad (3)$$

# Redes generativas adversárias

- Note que  $\theta^{(G)}$  somente aparece na segunda parcela de (3). Portanto, podemos ignorar a primeira parcela ao otimizar essa expressão e usando apenas os dados  $\mathbf{Z}$ . A solução que minimiza as duas perdas individualmente é a **solução minimax**; para o jogador G será aquela que minimiza a perda  $J^{(G)}$  sobre os parâmetros  $\theta^{(G)}$  enquanto maximiza esta perda sobre  $\theta^{(D)}$  e procedimento similar para  $J^{(D)}$ .
- Um método iterativo usa otimização via gradiente descendente estocástico e este pode ou não convergir. Estudos mostram que GANs podem produzir bons resultados em dados com baixas variabilidade e resolução.
- Várias variantes do GAN surgiram para melhorá-lo:
  - DCGAN: usa redes convolucionais profundas;
  - cGAN, ACGAN: conditional GAN
  - WGAN, WGAN-GP: diferentes funções perda.
- Para detalhes veja Assunção (2022).

## Redes generativas adversárias - Exemplo

- Neste exemplo, voltamos a usar os dados Mnist e uma versão do GAN, a ACGAN. Um código em R pode ser encontrado em

[https://github.com/rstudio/keras/blob/main/vignettes/examples/mnist\\_acgan.R](https://github.com/rstudio/keras/blob/main/vignettes/examples/mnist_acgan.R)

- Vemos, abaixo, a última iteração do algoritmo, indicando as perdas do discriminador, 1,3604, e a do gerador, 0,9898.

Epoch 20/20

235/235 [=====] - 203s 863ms/step - d\_loss: 1.3604 - g\_loss: 0.9898

- Na Figura 5 notamos que, enquanto a função perda do discriminador diminui ao longo das épocas, o mesmo não acontece com a perda do gerador. Na Figura 6 temos o dígito 6 previsto corretamente pelo modelo.

## Redes generativas adversárias - Exemplo

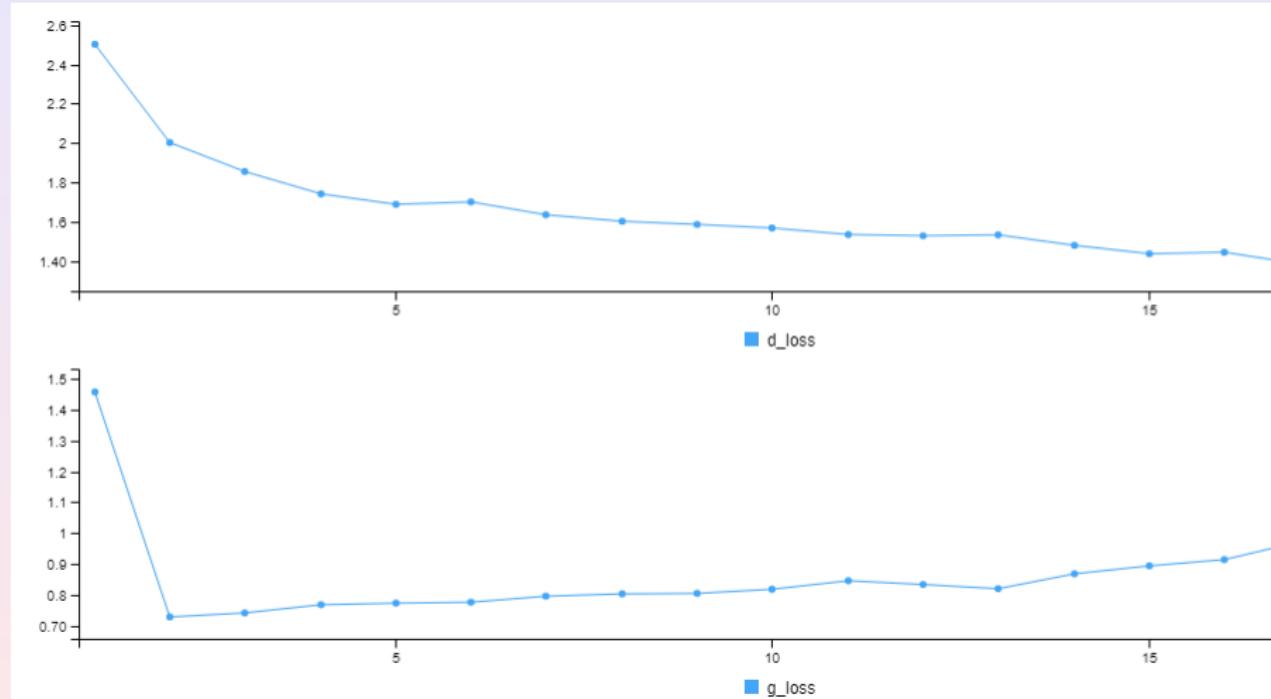


Figura 5: Comportamento das perdas do gerador e discriminador para o Exemplo.

## Redes generativas adversárias - Exemplo

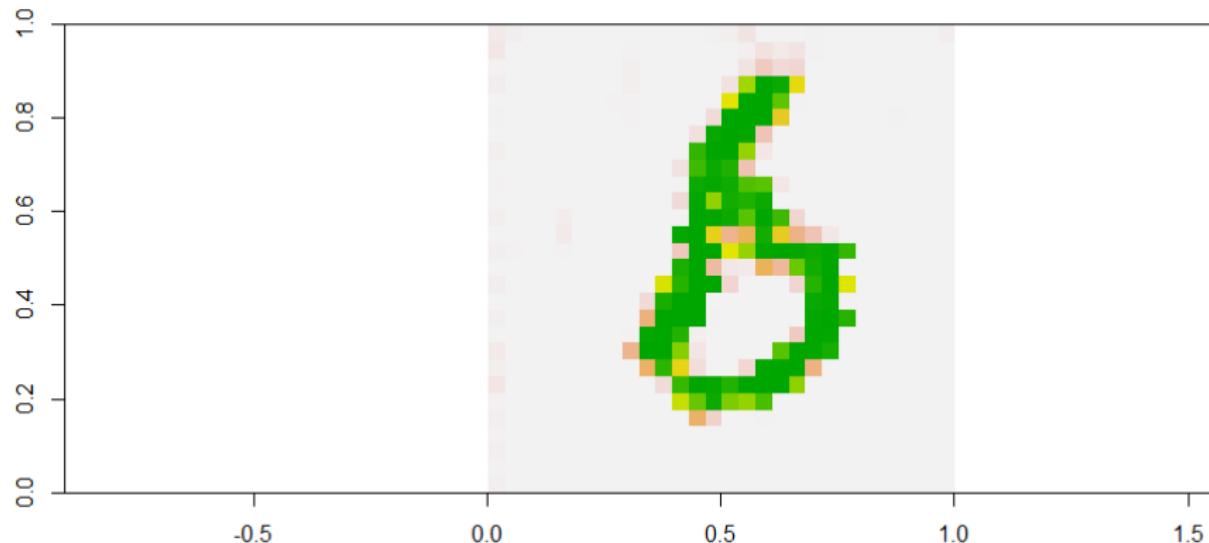


Figura 6: Dígito 6 previsto pelo modelo.

## Referências

- Assunção, R. (2022). Deep Learning. Short Course, SINAPE 2022, Gramado, RS.
- Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. and Bengio, Y. (2014). Generative adversarial networks, arXiv:1406.2661.
- Morettin, P. A. and Singer, J. M. (2023). *Estatística e Ciência de Dados*. 2a. edição. LTC.

# MAE 5905: Introdução à Ciência de Dados

Pedro A. Morettin

Instituto de Matemática e Estatística  
Universidade de São Paulo  
[pam@ime.usp.br](mailto:pam@ime.usp.br)  
<http://www.ime.usp.br/~pam>

## Aula 17

26 de junho de 2023

# Sumário

## 1 Simulação estática

## 2 Métodos de reamostragem

## Preliminares

- Sabemos como construir alguns modelos probabilísticos para representar uma situação real, ou então para descrever um experimento aleatório. Notamos, também, que determinados um espaço amostral e probabilidades associadas aos pontos desse espaço, o modelo probabilístico ficará completamente determinado e poderemos, então, calcular a probabilidade de qualquer evento aleatório.
- Muitas vezes, mesmo construindo um modelo probabilístico, certas questões não podem ser resolvidas analiticamente e teremos que recorrer a **estudos de simulação** para obter aproximações de quantidades de interesse.
- Estudos de simulação tentam reproduzir num ambiente controlado o que se passa com um problema real. Para nossos propósitos, a solução de um problema real consistirá na simulação de **variáveis aleatórias** (**simulação estática**) ou de **processos estocásticos** (**simulação dinâmica**).
- A simulação de variáveis aleatórias deu origem aos chamados **métodos Monte Carlo** (MMC), que por sua vez supõem que o pesquisador disponha de um **gerador de números aleatórios**.

## Preliminares

- Um **número aleatório** (NA) representa o valor de uma variável aleatória uniformemente distribuída no intervalo  $(0, 1)$ . Originalmente, esses NA eram gerados manualmente ou mecanicamente, usando dados, roletas etc. Modernamente, usamos computadores para gerar NA.
- Os MMC apareceram durante a segunda guerra mundial , em pesquisas relacionadas à difusão aleatória de neutrons num material radioativo. Os trabalhos pioneiros devem-se a Metropolis e Ulam (1949), Metropolis et al. (1953) e von Neumann (1951). Veja Sóbol (1976), Hammersley e Handscomb (1964) e Ross (1997).
- Para ilustrar, suponha que se queira calcular a área da figura F contida no quadrado Q de lado unitário (Figura 1). Suponha que sejamos capazes de gerar pontos aleatórios em Q, de modo homogêneo, isto é, de modo a cobrir toda a área do quadrado, ou ainda, que estes pontos sejam *uniformemente distribuídos sobre Q*. Se gerarmos  $N$  pontos, suponha que  $N'$  desses caiam em F. Então, poderemos aproximar a área de F por  $N'/N$ . Quanto mais pontos gerarmos, melhor será a aproximação.
- Note que o problema em si não tem nenhuma componente aleatória: queremos calcular a área de uma figura plana. Mas, para resolver o problema, uma possível maneira foi considerar um mecanismo aleatório. Veremos que esse procedimento pode ser utilizado em muitas situações.

# Preliminares

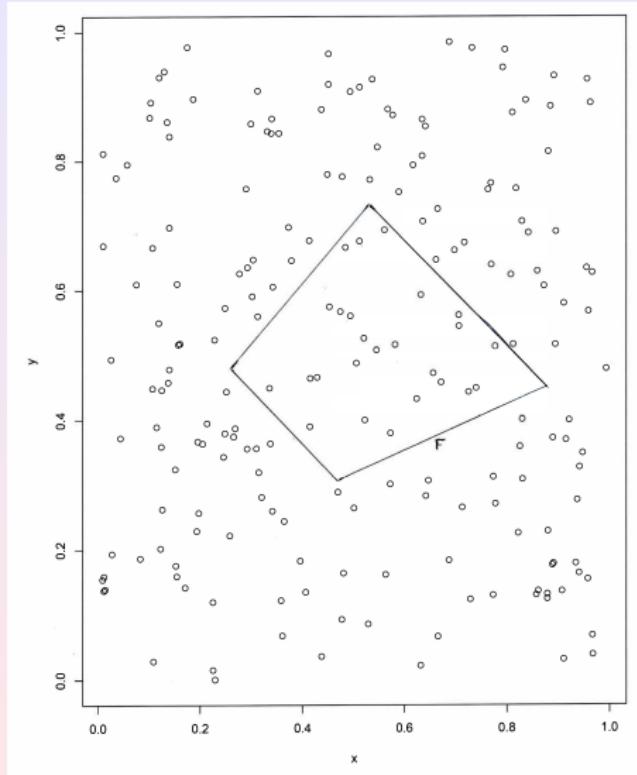


Figura: Área de uma figura por simulação.

## Preliminares

- Dissemos acima que um NA é um valor de uma variável aleatória uniformemente distribuída no intervalo  $(0, 1)$ . Vejamos algumas maneiras de obter um NA.
- **Exemplo 1.** Lance uma moeda 3 vezes e atribua o valor 1 se ocorrer cara e o valor 0 se ocorrer coroa. Os resultados possíveis são as *sequências* ou **números binários** abaixo:

000, 001, 010, 011, 100, 101, 110, 111.

- Cada um desses números binários corresponde a um número decimal. Por exemplo,  $(111)_2 = (7)_{10}$ , pois  $(111)_2 = 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$ . Considere a representação decimal de cada sequência acima e divida o resultado por  $2^3 - 1 = 7$ . Obteremos os números aleatórios  $0, 1/7, 2/7, \dots, 1$ . Qualquer uma das 8 sequências acima tem a mesma probabilidade, a saber,  $1/2^3 = 1/8$ .
- De modo geral, lançando-se a moeda  $n$  vezes, teremos  $2^n$  possibilidades e cada uma terá probabilidade  $1/2^n$  e os NA finais são obtidos por meio de divisão por  $2^n - 1$ .

## Preliminares

- O que se faz atualmente é fazer **simulação por meio de computadores**, que utiliza **números pseudo-aleatórios**, no lugar de NA.
- Os números pseudo-aleatórios (NPA) são obtidos por meio de técnicas que usam relações matemáticas recursivas **determinísticas**. Logo, um NPA gerado numa iteração dependerá do número gerado na iteração anterior, e portanto não será realmente aleatório, donde o nome pseudo-aleatório.
- Um método bastante utilizado em pacotes computacionais é o método congruencial
- O R usa o comando **runif(n,min,max)**, onde *n* é o número de valores a gerar e (*min*, *max*) é o intervalo no qual se quer gerar os NPA. No nosso caso, *min*=0 e *max*=1.
- Outros pacotes têm seus próprios geradores de NPA, como o MatLab.
- **Exemplo 2.** O comando  $u \leftarrow \text{runif}(10,0,1)$  pede para gerar 10 NA e guardá-los no vetor *u*.

```
> u<-runif(10,0,1)
```

```
> u
```

```
[1] 0.80850094 0.56611676 0.75882010 0.89910843 0.48447125  
[6] 0.02119849 0.06239355 0.30022882 0.12722598 0.49714446
```

# Preliminares

Existem três grandes grupos de métodos de simulação:

- 1) **Métodos de Simulação Estática.** Aqui, os procedimentos têm por objetivo gerar amostras independentes. Citamos os métodos Monte Carlo, aceitação/rejeição e reamostragem ponderada.
- 2) **Métodos de Simulação por Imputação.** A ideia desses métodos é aumentar os dados, introduzindo **dados latentes**, com o intuito de facilitar a simulação. Dentre esses métodos citamos o algoritmo EM ( de *expectation-maximization*) e o algoritmo de dados aumentados.
- 3) **Métodos de Simulação Dinâmica.** Esses métodos são denominados atualmente por MCMC (**Markov Chain Monte Carlo**) e têm por objetivo construir uma cadeia de Markov, cuja distribuição de equilíbrio seja a distribuição da qual queremos amostrar. Os métodos mais importantes aqui são o amostrador de Gibbs e os algoritmos de Metropolis e Metropolis-Hastings.

## Métodos Monte Carlo

- Suponha uma v.a. com distribuição  $F$  e desejamos calcular a média de uma função qualquer  $h(X)$ . Suponha, ainda, que exista um método para simular uma amostra  $X_1, \dots, X_n$  de  $F$ . Nas seções seguintes veremos alguns desses métodos. Então, o método Monte Carlo (MMC) consiste em aproximar  $\mu_F = E_F[h(X)]$  por

$$\hat{\mu}_F = \hat{E}_F[h(X)] = \frac{1}{n} \sum_{i=1}^n h(X_i). \quad (1)$$

- Observe que (14) aproxima a integral  $\int h(x)dF(x)$  ou  $\int h(x)f(x)dx$ , se existir a densidade de  $X$ . A lei (forte) dos grandes números garante que, quando  $n \rightarrow \infty$ ,  $\hat{\mu}_F$  converge para  $\mu_F$  com probabilidade um.
- O erro padrão da estimativa (14) é dado pela raiz quadrada da variância de  $\hat{\mu}_F$ , denotada por  $\text{Var}(\hat{\mu}_F)$ . Esta, por sua vez, pode ser estimada por  $\widehat{\text{Var}}(\hat{\mu}_F)$  e, portanto, uma estimativa do erro padrão de  $\hat{\mu}_F$  será

$$\widehat{\text{EP}}(\hat{\mu}_F) = \frac{1}{\sqrt{n}} \left[ \sum_{i=1}^n (h(X_i) - \hat{\mu}_F)^2 \right]^{1/2} = O(n^{-1/2}). \quad (2)$$

## MMC – Exemplo 3

- **Exemplo 3.** Suponha que se queira calcular o valor esperado de  $h(X)$ , onde  $h(x) = \sqrt{1 - x^2}$  e  $F \sim \mathcal{U}(0, 1)$ . Então, se  $X_1, \dots, X_n$  for uma amostra da distribuição uniforme padrão,

$$\hat{E}_F[h(X)] = \frac{1}{n} \sum_{i=1}^n \sqrt{1 - X_i^2}.$$

- Por exemplo, gerando-se 1.000 valores de uma  $\mathcal{U}(0, 1)$ , obtivemos o valor 0,7880834. Observe que essa é também, uma estimativa de um quarto da área de um círculo unitário, ou seja,  $\pi/4 = 0,7853982$ . O erro padrão calculado por (2) é 0,0069437.
- Uma outra aplicação do MMC é na obtenção de amostras de distribuições marginais. Suponha, por exemplo, que as v.a.  $X$  e  $Y$  tenham densidade conjunta  $f(x, y)$  e marginais  $f_X(x)$  e  $f_Y(y)$ , respectivamente. Então,

$$f_Y(y) = \int_{-\infty}^{\infty} f(x, y) dx = \int_{-\infty}^{\infty} f_X(x) f_{Y|X}(y|x) dx, \quad (3)$$

onde  $f_{Y|X}(y|x)$  é a densidade condicional de  $Y$  dado que  $X = x$ .

## MMC – Exemplo 3

Para obter uma amostra de  $f_Y(y)$  procedemos como segue (**método da composição ou mistura**):

- obtemos um elemento  $x^*$  de  $f_X(x)$ ;
- fixado  $x^*$ , obtemos um elemento  $y^*$  de  $f_{Y|X}(y|x^*)$ .

Repetimos os passos (a) e (b)  $n$  vezes, obtendo-se  $(x_1, y_1), \dots, (x_n, y_n)$  como uma amostra de  $f(x, y)$ , enquanto que  $y_1, \dots, y_n$  representa uma amostra de  $f_Y(y)$ .

É óbvio que devemos saber como amostrar das densidades  $f_X(x)$  e  $f_{Y|X}(y|x)$ ;  $x^*$  é chamado o **elemento misturador**.

## Simulação de variáveis discretas

- Vimos que a geração de NAs corresponde a gerar valores de uma distribuição uniforme no intervalo  $(0, 1)$
- Se  $U \sim \mathcal{U}(0, 1)$  e se  $0 < a < b < 1$ , então

$$P(a < U < b) = b - a. \quad (4)$$

- Considere, agora, uma v.a. qualquer  $X$ , com a distribuição de probabilidades dada abaixo:

$$\begin{aligned} X &: & x_1, & x_2, & \dots, & x_n \\ p_j &: & p_1, & p_2, & \dots, & p_n \end{aligned}$$

## Simulação de variáveis discretas

- Geramos, agora, um NA  $u$ ; Coloque:

$$X = \begin{cases} x_1, & \text{se } u < p_1, \\ x_2, & \text{se } p_1 \leq u < p_1 + p_2, \\ \dots \\ x_j, & \text{se } p_1 + \dots + p_{j-1} \leq u < p_1 + \dots + p_j. \end{cases} \quad (5)$$

- Como

$$P(X = x_j) = P(p_1 + \dots + p_{j-1} \leq U < p_1 + \dots + p_j) = p_j,$$

usando (4), vemos que  $X$  tem a distribuição que queremos.

- **Exemplo 4. Simulação de Uma Distribuição de Bernoulli.**
- Suponha que  $X$  tenha uma distribuição de Bernoulli, com  $P(X = 0) = 1 - p = 0,48$  e  $P(X = 1) = p = 0,52$ . Para gerar valores de tal distribuição basta gerar NA  $u$  e concluir:
  - Se  $u < 0,48$ , coloque  $X = 0$ ;

Se  $u \geq 0,48$ , coloque  $X = 1$ .

# Simulação de variáveis discretas

- Por exemplo, suponha que geramos dez NA : 0, 11; 0, 82; 0, 00; 0, 43; 0, 56; 0, 60; 0, 72; 0, 42; 0, 08; 0, 53. Então, os dez valores gerados da distribuição em questão são 0, 1, 0, 0, 1, 1, 1, 0, 0, 1, respectivamente.
- Exemplo 5. Simulação de Uma Distribuição Binomial.**
- Sabemos que se  $Y \sim b(n, p)$ , então  $Y$  é o número de sucessos num experimento de Bernoulli, com  $n$  repetições, e probabilidade de sucesso  $p$ .
- No Exemplo 4, obtivemos 5 sucessos, logo  $Y = 5$ . Portanto, se  $Y \sim b(10; 0,52)$ , e quisermos, digamos, gerar 20 valores dessa distribuição, basta considerar 20 experimentos de Bernoulli, sendo que em cada um deles repetimos o experimento  $n = 10$  vezes, com probabilidade de sucesso  $p = 0,52$ .
- Para cada experimento  $j$  consideramos o número de sucessos (número de 1),  $y_j$ ,  $j = 1, 2, \dots, 20$ . Obteremos, então, os 20 valores simulados  $y_1, \dots, y_{20}$  da v.a.  $Y$ . Observe que esses valores serão inteiros entre 0 e 20, inclusive esses dois valores.

## Simulação de variáveis discretas

**Exemplo 6. Simulação de Uma Distribuição de Poisson.**

Se  $N \sim P(\lambda)$ , então  $P(N = j) = p_j$  é dada por

$$P(N = j) = \frac{e^{-\lambda} \lambda^j}{j!}, \quad j = 0, 1, \dots \quad (6)$$

- A geração de valores de uma distribuição de Poisson parte da seguinte relação recursiva, que pode ser facilmente verificada:

$$p_{j+1} = \frac{\lambda}{j+1} p_j, \quad j \geq 0. \quad (7)$$

- Seja  $F(j) = P(N \leq j)$  a função de distribuição acumulada (f.d.a.) de  $N$ .
- Considere  $j$  o valor atual gerado e queremos gerar o valor seguinte. Chamemos simplesmente  $p = p_j$  e  $F = F(j)$ . Então o algoritmo para se gerar os sucessivos valores é o seguinte:

## Simulação de variáveis discretas

Passo 1: Gere o NA  $u$ ;

Passo 2: Faça  $j = 0$ ,  $p = e^{-\lambda}$  e  $F = p$ ;

Passo 3: Se  $u < F$ , coloque  $N = j$ ;

Passo 4: Faça  $p = \frac{\lambda}{j+1}p$ ,  $F = F + p$  e  $j = j + 1$ ;

Passo 5: Volte ao Passo 3.

Note que, no Passo 2, se  $j = 0$ ,  $P(N = 0) = p_0 = e^{-\lambda}$  e  $F(0) = P(N \leq 0) = p_0$ .

## Simulação de variáveis discretas

Suponha, por exemplo, que queiramos simular valores de uma distribuição de Poisson com parâmetro  $\lambda = 2$ . Então  $e^{-2} = 0,136$ . Obtemos:

Passo 1: Geramos  $u = 0,35$ ;

Passo 2:  $j = 0$ ,  $p = 0,136$ ,  $F = 0,136$ ;

Passo 3:  $u > F$ ;

Passo 4:  $p = 2(0,136) = 0,272$ ,  $F = 0,136 + 0,272 = 0,408$ ,  $j = 1$ ;

Passo 5: voltemos ao Passo 3; com  $u < F$ , colocamos  $N = 1$ . Temos, portanto o primeiro valor gerado da distribuição. Prosseguimos com o algoritmo para gerar outros valores.

# Simulação de variáveis discretas

O R e a planilha Excel possuem subrotinas próprias para simular valores de uma dada distribuição de probabilidades. A Tabela 1 traz as distribuições discretas contempladas por cada um e os comandos apropriados.

Tabela 1- Opções de Distribuições Discretas

| Distribuição    | Excel(Par.)                | R(Par.)                   |
|-----------------|----------------------------|---------------------------|
| Bernoulli       | <code>Bernoulli(p)</code>  | –                         |
| Binomial        | <code>Binomial(n,p)</code> | <code>binom(n,p)</code>   |
| Geométrica      | –                          | <code>geom(p)</code>      |
| Hipergeométrica | –                          | <code>hyper(N,r,k)</code> |
| Poisson         | <code>Poisson(λ)</code>    | <code>pois(λ)</code>      |

## Simulação de variáveis contínuas

- Considere uma v.a.  $X$ , com função de distribuição acumulada  $F$ , representada na Figura 2. Usando-se um gerador de NA, obtemos um valor  $u$ . Marca-se esse valor no eixo das ordenadas e por meio da função inversa de  $F$  obtém-se o valor  $x$  da v.a.  $X$  no eixo das abscissas. Ou seja, estamos resolvendo a equação

$$F(x) = u, \quad (8)$$

ou  $x = F^{-1}(u)$ . Formalmente, estamos usando o **método da transformação integral**, consubstanciada no seguinte resultado. Suponha  $F$  estritamente crescente.

- **Proposição.** Se  $X$  for uma v.a. com f.d.a  $F$ , então a v.a.  $U = F(X)$  tem distribuição uniforme no intervalo  $[0, 1]$ .
- O resultado pode ser estendido para o caso de  $F$  ser não decrescente, usando-se uma definição mais geral de inversa.

## Simulação de variáveis contínuas

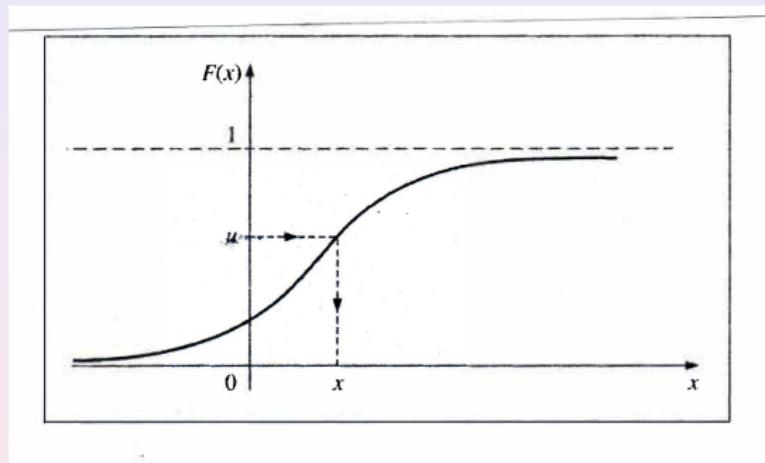


Figura: Função de distribuição acumulada de uma v.a.  $X$ .

## Simulação de variáveis contínuas

## • Exemplo 7. Simulação de uma Distribuição Exponencial.

Se a v.a.  $T$  tiver densidade dada por

$$f(t) = \frac{1}{\beta} e^{-t/\beta}, \quad t > 0, \tag{9}$$

a sua f.d.a. é dada por

$$F(t) = 1 - e^{-t/\beta}, \tag{10}$$

logo temos que resolver esta equação para gerar  $t$ .

• Tomando logaritmo na base  $e$ , temos

$$1 - u = e^{-t/\beta} \Leftrightarrow \log(1 - u) = -\frac{t}{\beta} \Leftrightarrow t = -\beta \log(1 - u).$$

Logo, gerado um NA, um valor da distribuição  $\text{Exp}(\beta)$  é dado por  $-\beta \log(1 - u)$ .

## Simulação de variáveis contínuas

- Por exemplo, suponha  $\beta = 2$  e queremos gerar 5 valores de  $T \sim \text{Exp}(2)$ . Gerados os valores

$u_1 = 0,57, u_2 = 0,19, u_3 = 0,38, u_4 = 0,33, u_5 = 0,31$  de uma distribuição uniforme em  $(0, 1)$  (os números aleatórios), obteremos

$$t_1 = (-2)(\log(0,43)) = 1,68, \quad t_2 = (-2)(\log(81)) = 0,42,$$

$$t_3 = (-2)(\log(0,62)) = 0,96, \quad t_4 = (-2)(\log(0,67)) = 0,80,$$

$$t_5 = (-2)(\log(0,69)) = 0,74.$$

- Podemos reduzir um pouco os cálculos se usarmos o seguinte fato: se  $U \sim \mathcal{U}(0, 1)$ , então  $1 - U \sim \mathcal{U}(0, 1)$ . Resulta que poderemos gerar os valores de uma exponencial por meio de

$$t = -\beta \log(u).$$

- Usando esta fórmula para os valores de  $U$  acima, obteremos os seguintes valores de  $T$  : 1,12; 3,32; 1,93; 0,96; 2,34.

## Simulação de variáveis contínuas

## • Exemplo 8. Simulação de uma Distribuição Normal.

Há vários métodos para gerar v.a. normais, mas uma observação importante é que basta gerar uma v.a. normal padrão, pois qualquer outra pode ser obtida desta. De fato, gerado um valor  $z_1$  da v.a.  $Z \sim N(0, 1)$ , para gerar um valor de uma v.a.  $X \sim N(\mu, \sigma^2)$  basta usar a transformação  $z = (x - \mu)/\sigma$  para obter

$$x_1 = \mu + \sigma z_1. \quad (11)$$

- Um método usa a transformação integral e uma tabela de probabilidades para a normal padrão.
- Vejamos um exemplo. Suponha que  $X \sim N(10; 0, 16)$ , ou seja,  $\mu = 10$  e  $\sigma = 0, 4$ . Temos que resolver a equação (7), ou seja,

$$\Phi(z) = u,$$

onde estamos usando a notação  $\Phi(z)$  para a f.d.a. da  $N(0, 1)$ .

- Por exemplo, gerado um NA  $u = 0, 230$ , temos que resolver

$$\Phi(z) = 0, 230,$$

ou seja, temos que encontrar o valor  $z$  tal que a área à sua esquerda, sob a curva normal padrão, seja 0, 230. Veja a Figura 3.

## Simulação de variáveis contínuas

Consultando uma tabela para a normal, encontramos que  $z = -0,74$ . Logo o valor gerado da normal em questão satisfaz

$$\frac{x - 10}{0,4} = -0,74,$$

ou seja,  $x = 10 + (0,4)(-0,74) = 9,704$ . Qualquer outro valor pode ser gerado da mesma forma.

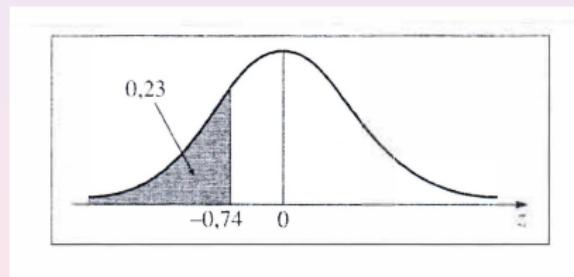


Figura: Geração de um valor  $z \sim N(0, 1)$ .

# Simulação de variáveis contínuas

- Há outros métodos mais eficientes. Alguns são variantes do método de Box - Müller (1958).
- Nesse método, são geradas duas v.a.  $Z_1$  e  $Z_2$ , independentes e  $N(0, 1)$ , por meio das transformações

$$\begin{aligned} Z_1 &= \sqrt{-2 \log U_1} \cos(2\pi U_2), \\ Z_2 &= \sqrt{-2 \log U_1} \sin(2\pi U_2), \end{aligned} \tag{12}$$

em que  $U_1$  e  $U_2$  são v.a. com distribuição uniforme em  $[0, 1]$ .

- Portanto, basta gerar dois NAs  $u_1$  e  $u_2$  e depois gerar  $z_1$  e  $z_2$  usando (12).
- O método de Box-Müller pode ser computacionalmente ineficiente, pois necessita calcular senos e cossenos. Uma variante, chamado de método polar, evita esses cálculos.

## Simulação de variáveis contínuas

Com o R podemos usar o comando `qnorm`, para obter um quantil de uma distribuição normal, a partir de sua f.d.a. Por exemplo, para gerar 1.000 valores de uma distribuição normal padrão, usamos:

```
u <- runif(1000,0,1) # gera 1000 observações de uma uniforme[0,1]
x <- qnorm(u,mean=0, sd = 1) # Calcula os quantis para o vetor
                                u simulado da uniforme
```

```
par(mfrow=c(1,2))
hist(u, freq=FALSE, main="Histograma da amostra da distribuição
                            Uniforme simulada")
hist(x, freq=FALSE, main="Histograma da variável X simulada
                            a partir do resultado do Teorema 15.1")
```

Os histogramas, da uniforme e da normal, estão na Figura 4.

# Simulação de variáveis contínuas

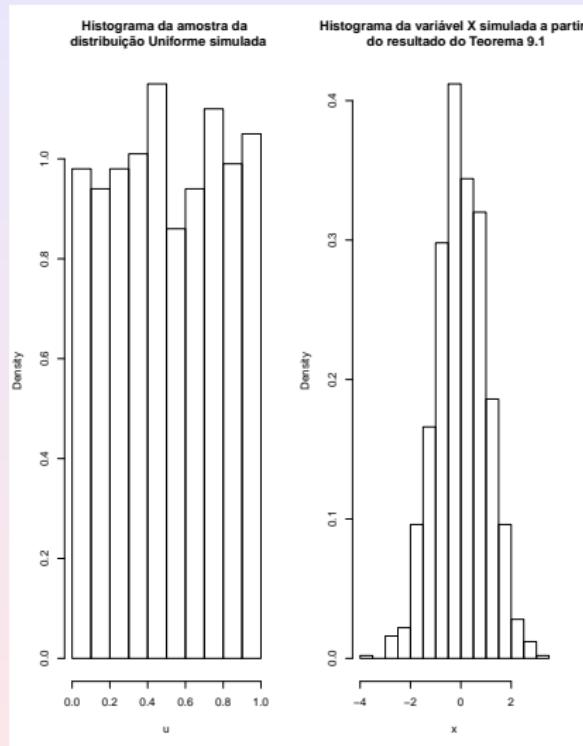


Figura: Simulação de uma normal padrão via R.

# Simulação de variáveis contínuas

O R e a planilha Excel têm subrotinas próprias para gerar muitas das distribuições estudadas nesta seção. A Tabela 2 mostra as opções disponíveis e os comandos apropriados. Além da  $N(0,1)$  o Excel usa a função INV para gerar algumas outras distribuições contínuas.

Tabela 2: Opções de Distribuições Contínuas

| Distribuição | Excel(Par.) | R(Par.)                       |
|--------------|-------------|-------------------------------|
| Normal       | Normal(0,1) | $\text{norm}(\mu, \sigma)$    |
| Exponencial  | –           | $\text{exp}(\beta)$           |
| t(Student)   | –           | $t(\nu)$                      |
| F(Snedecor)  | –           | $f(\nu_1, \nu_2)$             |
| Gama         | –           | $\text{gamma}(\alpha, \beta)$ |
| Qui-Quadrado | –           | $\text{chisq}(\nu)$           |
| beta         | –           | $\text{beta}(\alpha, \beta)$  |

# Simulação de vetores aleatórios

- É mais complicado simular distribuições bidimensionais. No caso de  $X$  e  $Y$  serem **independentes**, então

$$f(x, y) = f_X(x)f_Y(y), \quad \forall x, y,$$

se elas forem contínuas, por exemplo. Logo, para gerar um valor  $(x, y)$  da densidade conjunta  $f(x, y)$ , basta gerar a componente  $x$  da distribuição marginal de  $X$  e a componente  $y$  da distribuição marginal de  $Y$ , **independentemente**.

- No caso de v.a. **dependentes**, temos que vale a relação:

$$f(x, y) = f_X(x)f_{Y|X}(y|x).$$

- Logo, por essa relação, primeiramente geramos um valor  $x$  da distribuição marginal de  $X$  e fixado esse valor,  $x_0$ , digamos, geramos um valor da distribuição condicional de  $X$ , dado que  $X = x_0$ . Isso implica que devemos saber como gerar valores das distribuições  $f_X(x)$  e  $f_{Y|X}(y|x)$ .

## Simulação de vetores aleatórios

**Exemplo 8. Distribuição Normal Bidimensional.**

O método de Box-Müller gera valores de duas normais padrões independentes,  $Z_1$  e  $Z_2$ . Logo, se quisermos gerar valores da distribuição conjunta de  $X$  e  $Y$ , **independentes e normais**, com  $X \sim N(\mu_x, \sigma_x^2)$  e  $Y \sim N(\mu_y, \sigma_y^2)$ , basta considerar

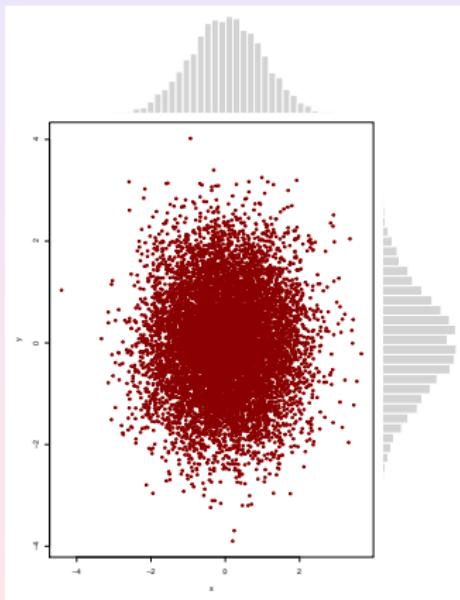
$$X = \mu_x + \sigma_x Z_1, \quad Y = \mu_y + \sigma_y Z_2.$$

O código em R, para o caso de duas normais padões, seria:

```
u1<-runif(10000,0,1)
u2<-runif(10000,0,1)
P<-data.frame(u1,u2)
x<-qnorm(u1)
y<-qnorm(u2)
```

## Simulação de vetores aleatórios

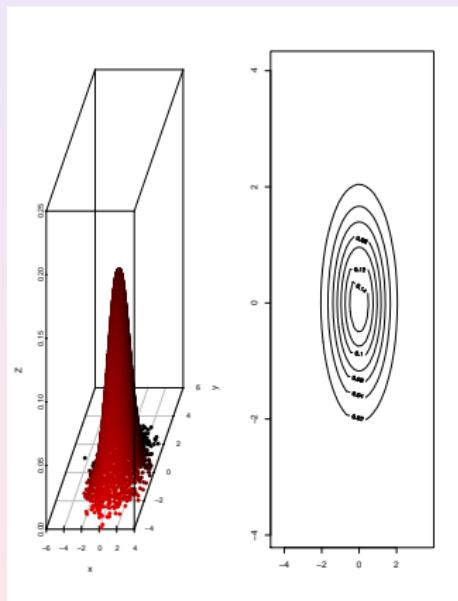
Na Figura 5 temos os histogramas das duas curvas juntamente com o diagrama de dispersão bidimensional obtidos gerando-se 10.000 valores cada uma dessas normais padrões independentes.



**Figura:** Simulação de duas normais independentes (nas margens) e gráfico de dispersão.

## Simulação de vetores aleatórios

Na Figura 6 temos a densidade bidimensional normal padrão e as respectivas curvas de nível.



**Figura:** Distribuição normal padrão bidimensional gerada e curvas de nível.

## Métodos de reamostragem

- Suponha que se queira simular uma amostra da densidade  $\pi$ , mas que isso não seja fácil. O que se pode fazer é proceder em duas etapas.
- Suponha que haja uma densidade  $g$ , da qual seja fácil gerar valores e que esteja próxima de  $\pi$ . Então:
  - (a) na primeira etapa simulamos uma amostra de  $g$ ;
  - (b) na segunda etapa, exercemos um mecanismo de correção, de modo que a amostra de  $g$  seja “direcionada” para tornar-se uma amostra de  $\pi$ .
- Em geral o que se faz é, ao simular um valor de  $g$ , este é aceito com certa probabilidade  $p$  e escolhendo-se  $p$  adequadamente podemos assegurar que o valor aceito seja um valor de  $\pi$ .

## Aceitação – rejeição

- Nesta situação, supomos que existe uma constante finita conhecida  $A$ , tal que  $\pi(x) \leq Ag(x)$ , para todo  $x$ . Ou seja,  $Ag$  serve como um envelope para  $\pi$  (Figura 7).
- Algoritmo:
  - [1]] Simule  $x^*$  de  $g(x)$ ;
  - [2]] simule  $u$  de uma distribuição  $\mathcal{U}(0, 1)$ , independentemente de  $x^*$ ;
  - [3]] se  $u \leq \pi(x^*)/Ag(x^*)$ , então aceite  $x^*$  como gerada de  $\pi(x)$ ; caso contrário, volte ao item 1.

## Aceitação – rejeição

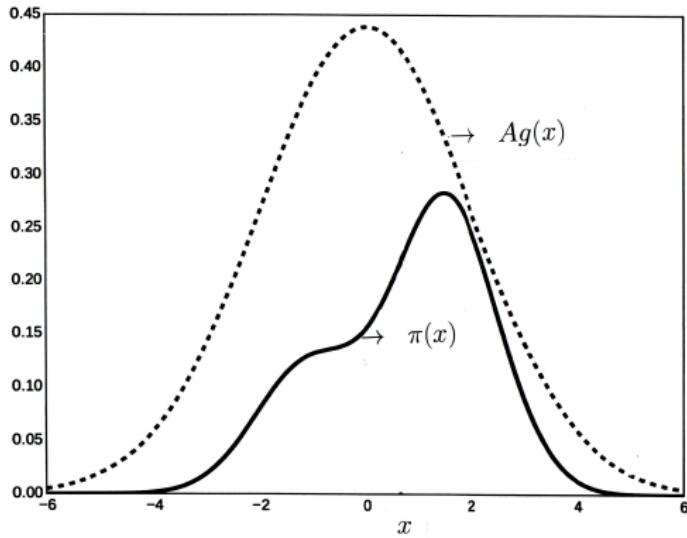


Figura: Densidades  $\pi$  (linha cheia) e  $Ag$  (linha pontilhada).

## Aceitação – rejeição

- Suponha que  $X \sim g(x)$ . Seja  $p(x)$  a probabilidade que  $x$  seja aceito; então,  $p(x) = \pi(x)/Ag(x)$ . Logo,

$$P(X \leq x \text{ e } X \text{ aceito}) = \int_{-\infty}^x p(y)g(y)dy$$

e

$$P(X \text{ aceito}) = \int_{-\infty}^{\infty} p(y)g(y)dy.$$

- Segue que

$$P(X \leq x \mid \text{aceito}) = \frac{\int_{-\infty}^x p(y)g(y)dy}{\int_{-\infty}^{\infty} p(y)g(y)dy} = \int_{\infty}^x \pi(y)dy,$$

logo os valores aceitos têm realmente distribuição  $\pi(x)$ .

- Também,

$$P(\text{aceitação}) = \int_{-\infty}^{\infty} p(y)g(y)dy = \frac{1}{A} \int_{-\infty}^{\infty} \pi(y)dy = \frac{1}{A}. \quad (13)$$

## Aceitação – rejeição

- Observe que  $\pi$  deve ser conhecida a menos de uma constante de proporcionalidade, ou seja, basta conhecer o que se chama o **núcleo** de  $\pi(x)$ .
- Devemos escolher  $g(x)$  de modo que ela seja facilmente simulável e de sorte que  $\pi(x) \approx Ag(x)$ , pois a chance de rejeição será menor. Também de (13), devemos ter  $A \approx 1$ .
- **Observações:** (a)  $0 < \pi(x^*)/Ag(x^*) \leq 1$ ;  
(b) O número de iterações,  $N$ , necessárias para gerar  $\pi$  é uma v.a. com distribuição geométrica, com probabilidade de sucesso

$$p = P(U \leq \pi(x^*)/Ag(x^*))P(N = n) = (1 - p)^{n-1}p, \quad n \geq 1.$$

Potanto, em média, o número de iterações necessárias é  $E(N) = 1/p$ .  
Como vimos acima,  $p = 1/A$ , logo  $E(N) = A$ . Logo, é desejável escolher  $g$  de modo que  $A = \sup_x \{\pi(x)/g(x)\}$ .

- (c) De (b), podemos dizer que o número esperado de iterações do algoritmo necessárias até que um valor de  $\pi$  seja gerado com sucesso é exatamente o valor da constante  $A$ .
- O método pode ser usado também para o caso de v.a.'s discretas.

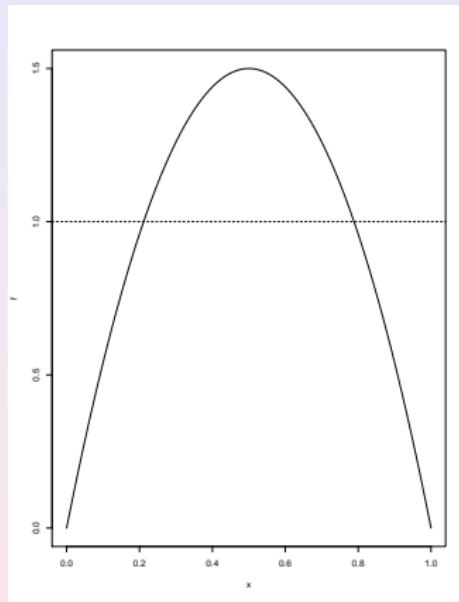
## Aceitação – rejeição

- **Exemplo 1.** Considere a densidade de uma Beta(2,2), ou seja,

$$\pi(x) = 6x(1-x), \quad 0 < x < 1.$$

- Suponha  $g(x) = 1$ ,  $0 < x < 1$ . O máximo de  $\pi(x)$  é 1,5, para  $x = 0,5$ .
- Logo, podemos tomar  $A = 1,5$  (o valor máximo de  $\pi(x)$ , para  $x = 0,5$ ), e teremos  $p = P(\text{aceitação}) = 1/A = 0,667$ .
- Portanto, para obter, por exemplo, uma amostra de tamanho 1.000 de  $\pi(x)$  deveremos simular em torno de 1.500 valores de uma uniforme padrão. Veja a Figura 8.

## Aceitação – rejeição



**Figura:** Densidades  $\pi$  (linha cheia) e  $g$  (linha tracejada) para o Exemplo 1.

## Aceitação – rejeição

Um algoritmo equivalente é o seguinte:

- 1) Simule  $x^*$  de  $g(x)$ ;
- 2) Simule  $y^*$  de  $\mathcal{U}(0, Ag(x^*))$ ;
- 3) Aceite  $x^*$  se  $y^* \leq \pi(x^*)$ ; caso contrário, volte a 1.

Usando o código R do capítulo, podemos obter as Figuras 9 e 10; a primeira mostra o histograma dos valores gerados (com a verdadeira curva adicionada) e a segunda mostra as regiões de aceitação e rejeição.

## Aceitação – rejeição

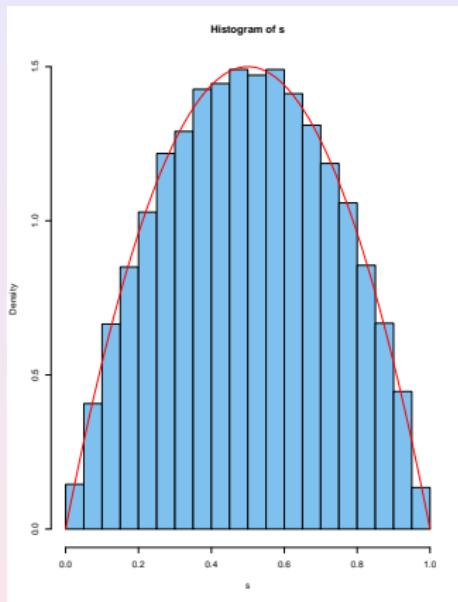
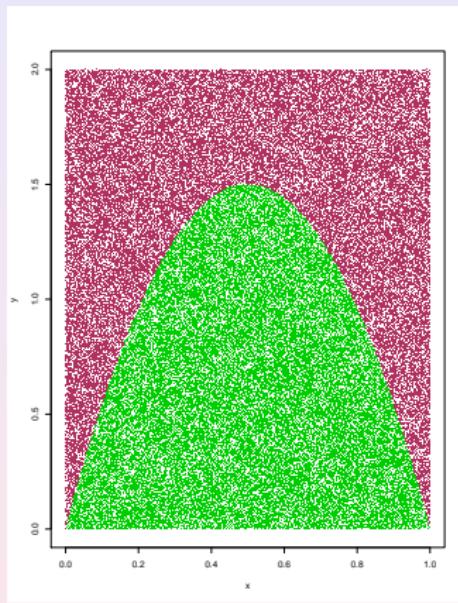


Figura: Histograma dos valores gerados e densidade (vermelho).

## Aceitação – rejeição



**Figura:** Região de aceitação (verde) e de rejeição (marrom).

# Reamostragem ponderada

Neste tipo de simulação temos essencialmente as duas etapas anteriores, mas  $Ag$  não precisa ser um envelope para  $\pi$ .

## Algoritmo:

- 1) Retire uma amostra  $x_1, \dots, x_n$  de  $g(x)$ ;
- 2) construa os pesos  $w_i$  dados por

$$w_i = \frac{\pi(x_i)/g(x_i)}{\sum_{j=1}^n \pi(x_j)/g(x_j)}, \quad i = 1, 2, \dots, n;$$

- 3) reamostre da distribuição de probabilidades discreta  $(x_i, w_i)$ ,  $i = 1, \dots, n$ .

## Reamostragem ponderada

- Então, a amostra resultante tem distribuição  $\pi$ . De fato, se  $x$  for um valor simulado pelo método,

$$F_x(a) = P(X \leq a) = \sum_{i:x_i \leq a} w_i = \frac{\sum_{i=1}^n (\pi(x_i)/g(x_i)) I_{\{x_i \leq a\}}}{\sum_{j=1}^n (\pi(x_j)/g(x_j))},$$

e o último termo converge, quando  $n \rightarrow \infty$  (lei forte dos grandes números), para

$$\frac{\int (\pi(x)/g(x)) I_{\{x \leq a\}} g(x) dx}{\int (\pi(x)/g(x)) dx} = \frac{\int \pi(x) I_{\{x \leq a\}} dx}{\int \pi(x) dx} = F_\pi(x).$$

- O método de reamostragem ponderada (**importance sampling**) é também usado para reduzir a variância da estimativa MC. Lembremos que o MMC consiste em aproximar  $E_F[h(X)]$  por

$$\hat{E}_F[h(X)] = \frac{1}{n} \sum_{i=1}^n h(X_i). \tag{14}$$

Suponha que em (14)  $F$  tenha densidade  $\pi$ . Então

$$\theta_\pi = E_\pi[h(X)] = \int h(x)\pi(x)dx = \int h(x)[\frac{\pi(x)}{g(x)}]g(x)dx. \tag{15}$$



## Reamostragem ponderada

- Se chamarmos  $\varphi(x) = h(x)\pi(x)/g(x)$ , teremos

$$\theta_\pi = \int \varphi(x)g(x)dx.$$

- Segue-se que, obtendo-se uma amostra  $x_1, \dots, x_n$  de  $g(x)$ , poderemos estimar (15) por

$$\hat{\theta}_\pi = \frac{1}{n} \sum_{i=1}^n \varphi(x_i) = \frac{1}{n} \sum_{i=1}^n w_i h(x_i), \quad (16)$$

onde  $w_i = \pi(x_i)/g(x_i)$ . Compare com o estimador MC de (14), que é não viesado, ao passo que (16) é viesado.

- Se quisermos um estimador não viesado, basta considerar

$$\theta_\pi^* = \frac{\sum_{i=1}^n w_i h(x_i)}{\sum_{i=1}^n w_i}. \quad (17)$$

- Note que o estimador dá mais peso a regiões onde  $g(x) < \pi(x)$ . Geweke (1989) provou que  $\theta_\pi^* \rightarrow \theta$ , com probabilidade um, se o suporte de  $g(x)$  inclui o suporte de  $\pi(x)$ ,  $X_i \sim \text{iid } g(x)$  e  $E[h(X)] < \infty$ . Mostrou, também, que o erro padrão da estimativa (17) é dado por

$$\frac{[\sum_{i=1}^n [h(x_i) - \theta_\pi^*]^2 w_i^2]^{1/2}}{\sum_{i=1}^n w_i}.$$

## Reamostragem ponderada

- **Exemplo 2.** Num modelo genético, 197 animais distribuem-se em quatro classes  $\mathbf{X} = (x_1, x_2, x_3, x_4)' = (125, 18, 20, 34)'$ , segundo as probabilidades  $(\theta + 2)/4, (1 - \theta)/4, (1 - \theta)/4, \theta/4$ , e o objetivo é estimar  $\theta$ .
- A verossimilhança é

$$L(\theta|\mathbf{X}) \propto (2 + \theta)^{125} (1 - \theta)^{38} \theta^{34}.$$

- Supondo uma priori constante , a densidade a posteriori é

$$p(\theta|\mathbf{X}) \propto (2 + \theta)^{125} (1 - \theta)^{38} \theta^{34}.$$

- Um estimador de  $\theta$  é a média dessa densidade a posteriori, que é estimada por (17). Aqui, suponha que  $g(\theta) \propto \theta^{34}(1 - \theta)^{38}$ ,  $0 < \theta < 1$ , ou seja, temos uma Beta (35,39).

## Reamostragem ponderada

- Portanto,

$$\hat{\theta} = \frac{\sum_{i=1}^n w_i \theta_i}{\sum_{i=1}^n w_i},$$

onde  $\theta_1, \dots, \theta_n$  é uma amostra de  $g(x)$  e os pesos  $w_i$  são dados por

$$w_i = \frac{(2 + \theta_i)^{125}}{\sum_{j=1}^n (2 + \theta_j)^{125}}, \quad i = 1, \dots, n.$$

Gerando-se 10.000 valores de uma Beta(35,39) obtivemos  $\hat{\theta} = 0,6180$ .

## Referências

- Box, G.E.P. and Müller, M.E.(1958). A note on the generation of random normal deviates. *The Annals of Statistics*, **29**, 610–611.
- Hammersley, J.M. and Handscomb, D.C. (1964). *Monte Carlo Methods*. New York: Wiley.
- Kleijnen, J. and Groenendall, W. (1994). *Simulation: A Statistical Perspective*. Chichester: Wiley.
- Ross, S.(1997). *Simulation*, 2nd Ed., New York: Academic Press.
- Sobol, I.M.(1976). *Método de Monte Carlo*. Moscow: Editorial MIR.
- von Neumann, J.(1951). Various techniques used in connection with random digits, Monte Carlo Method. *U.S. National Bureau of Standards Applied Mathematica Series*, **12**, 36–38.