

Machine Learning Aplicada à Agropecuária de Precisão

Dia 02 - Machine Learning Parte II



Giancarlo D. Salton, PhD

Applied Intelligence Research Centre & ADAPT Centre
School of Computing

Chapecó, 09 de maio de 2018

Decisões Fundamentadas nos Dados

Investigando os dados

Preparando os dados

Avaliando os modelos

Experimentando com os dados e modelos

Resumo

Decisões Fundamentadas nos Dados

- ▶ Um modelo será tão bom quanto as *features* usadas para aprendê-lo
- ▶ Entretanto, a maior dificuldade em *machine learning* é a *engenharia de features*
 - ▶ É por isso que precisamos conhecer o domínio do problema e explorar os dados que estão disponíveis

- ▶ Precisamos estar em sintonia com o domínio que estamos estudando
 - ▶ Qual é o problema que estamos tentando resolver?
 - ▶ Quais são os objetivos que pretendemos atingir?
 - ▶ Como funciona o domínio?
 - ▶ De que forma um modelo de IA pode ajudar a atingir os objetivos?

- ▶ Temos que analisar a viabilidade da solução
 - ▶ Temos acesso a *datasets* relativos ao domínio?
 - ▶ De que forma a solução será utilizada no processo de tomada de decisão?

- ▶ Aspectos que devemos considerar sobre os *datasets* e as *features*
 - ▶ Disponibilidade
 - ▶ Timing
 - ▶ Validade

► Aspectos legais também devem ser observados!

- Legislação contra discriminação
- Legislação de proteção aos dados

Investigando os dados

- ▶ Muitas vezes nos deparamos com problemas no nosso *dataset*
 - ▶ *Features* faltando
 - ▶ Erros de preenchimento
 - ▶ Valores altos ou baixos demais (*outliers*)

- ▶ No caso de encontrarmos problemas, podemos
 - ▶ remover a *feature* do *dataset*
 - ▶ realizar uma análise casuisticamente e remover apenas os *datapoints* problemáticos
 - ▶ designar um valor para a *feature* baseado nos demais valores da *feature*

- ▶ Quando decidimos designar um novo valor para a *feature*
 - ▶ se os valores são muito altos ou muito baixos, podemos definir “**limites**” → quaisquer valores maiores ou menores assumem automaticamente o mesmo valor do limite
 - ▶ se alguns valores estão faltando, podemos atribuir um novo baseado na mesma *feature* de outros *datapoints* → média, mediana, ...
 - ▶ um método menos comum é utilizar uma métrica de distância e encontrar os *datapoints* mais próximos daquele que contém problemas, designando assim novos valores baseados na proximidade

- ▶ É muito importante documentar o processo e as ações resultantes
 - ▶ devemos replicar o processo quando o sistema estiver em produção e com novos casos sendo submetidos para predição/classificação
 - ▶ no caso de questionamentos, a documentação pode ajudar a encontrar a resposta
 - ▶ experimentos

Preparando os dados

- ▶ Antes de “treinarmos” os modelos, precisamos ainda nos preocupar com a preparação dos dados
 - ▶ alguns algoritmos “requerem” um certo formato para os dados

► Normalização

- normalização por intervalo → $[mínimo, máximo]$

$$a'_i = \frac{a_i - \min(a)}{\max(a) - \min(a)} \times (\max(a) - \min(a)) + \min(a)$$

- normalização por z-score

$$a'_i = \frac{a_i - \bar{a}}{sd(a)}$$

► *Binning*

- ▶ transforma uma *feature* contínua em uma *feature* discreta
- ▶ define-se uma série de intervalos com divisão dos datapoints, atribuindo o número do intervalo como novo valor para a *feature*

- ▶ intervalos com o mesmo tamanho
- ▶ intervalos com a mesma quantia de *datapoints*

Avaliando os modelos

- ▶ Uma das coisas mais importantes na avaliação dos nossos modelos é **não utilizar o mesmo *dataset* usado para aprender o modelo**
- ▶ 3 razões para isso:
 - ▶ determinar qual o melhor modelo
 - ▶ obter uma estimativa de como o modelo vai se comportar em produção
 - ▶ convencer os usuários de que ele funciona

- ▶ Normalmente, deixamos de fora uma parte dos *datapoints* durante a etapa de aprendizado do modelo
 - ▶ Chamado “*dataset de teste*”
- ▶ Após o modelo ter sido aprendido, aplica-se o mesmo no “*dataset de teste*” e mede-se a performance com alguma métrica

- ▶ Temos que prestar atenção nos tipos de predições que o modelo está fazendo
 - ▶ Positivos verdadeiros (TP - *True Positive*)
 - ▶ Negativos verdadeiros (TN - *True Negative*)
 - ▶ Falsos positivos (FP - *False Positive*)
 - ▶ Falsos negativos (FN - *False Negative*)

		Classificação	
		positivo	negativo
Alvo	positivo	TP	FN
	negativo	FP	TN

$$\text{taxa de acertos} = \frac{(TP + TN)}{(TP + TN + FP + FN)}$$

$$\text{taxa de erros} = \frac{(FP + FN)}{(TP + TN + FP + FN)}$$

$$\text{precision} = \frac{TP}{(TP + FP)}$$

$$\text{recall} = \frac{TP}{(TP + FN)}$$

$$F_1\text{-score} = 2 \times \frac{(\text{precision} \times \text{recall})}{(\text{precision} + \text{recall})}$$

- ▶ Muito do *machine learning* envolve criar experimentos para avaliar modelos e combinações de *features*
 - ▶ seleção de *features*
 - ▶ métodos para lidar com informações faltando
 - ▶ métodos para lidar com informações errôneamente preenchidas
 - ▶ ...
- ▶ *Engenharia de features*

Experimentando com os dados e modelos

- ▶ Um componente significativo da prática do machine learning são os experimentos
- ▶ Experimentos possuem diversos objetivos
 - ▶ analisar qual o melhor modelo para o projeto
 - ▶ validar as decisões tomadas nas fases de exploração e preparação dos dados
 - ▶ redução do número de *features*
 - ▶ *fine-tunning* → vários algoritmos possuem parâmetros usados para construir os modelos e que podem ser otimizados
 - ▶ às vezes os resultados podem variar dramaticamente!
- ▶ Alguns desses objetivos podem ser estudados e alcançados numa mesma rodada de experimentos

- ▶ Muito do *machine learning* ainda funciona na base da tentativa e erro
- ▶ Não há como prever qual a melhor combinação de algoritmo, *features*, procedimentos de preparação, etc, antes de aplicá-los e testá-los
 - ▶ Por isso, a documentação do processo é extremamente importante!

- ▶ Para organizar uma rodada de experimentos, precisamos
 - ▶ Do(s) algoritmo(s) a ser(em) testado(s)
 - ▶ Do *dataset* já preparado
 - ▶ Definição da(s) métrica(s) a ser(em) avaliada(s)

- ▶ A forma mais simples de se criar um experimento é usando um *dataset de teste*
 - ▶ uma parte dos *datapoints* são deixados de lado na etapa de aprendizado do modelo

Training Set	Validation Set	Test Set
--------------	----------------	----------

(a) Divisão 50:20:30

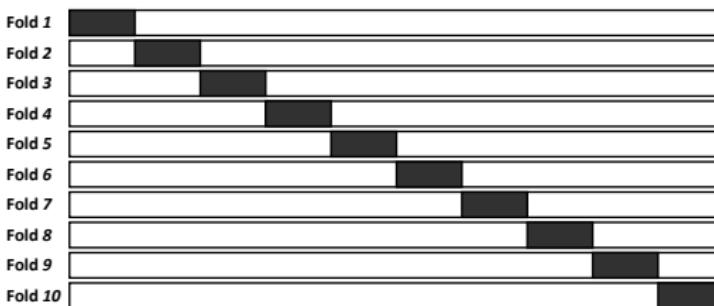
Training Set	Validation Set	Test Set
--------------	----------------	----------

(b) Divisão 40:20:40

Fonte: John D. Kelleher, Brian Mac Namee, and Aoife D'Arcy. 2015. *Fundamentals of Machine Learning for Predictive Data Analytics: Algorithms, Work Examples and Case Studies*. MIT Press.
<http://machinelearningbook.com/>

- ▶ Um problema que pode ocorrer com o procedimento de criação do *dataset de teste* é que os casos *fáceis* podem ter ficado nesta parte dos *datapoints*
 - ▶ performance do modelo será superestimada
 - ▶ quando colocado em produção o modelo terá uma péssima performance
- ▶ Solução: *validação cruzada*

- ▶ O método de *K-fold* é um procedimento iterativo para testar um algoritmo e o modelo gerado
 - ▶ a cada iteração um *dataset de teste* diferente é criado enquanto os *datapoints* restantes formam o *dataset de treino*
 - ▶ repete-se o processo *K* vezes para obter uma média das métricas de performance do modelo como resultado final
 - ▶ **normalmente o *dataset de teste* é gerado aleatoriamente**

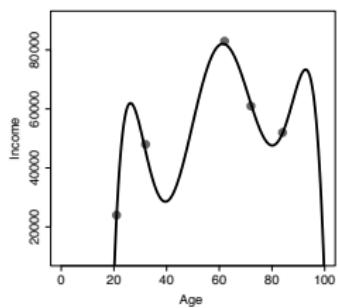


Fonte: John D. Kelleher, Brian Mac Namee, and Aoife D'Arcy. 2015. *Fundamentals of Machine Learning for Predictive Data Analytics: Algorithms, Work Examples and Case Studies*. MIT Press.
<http://machinelearningbook.com/>

- ▶ Alguns algoritmos possuem parâmetros que controlam o modelo aprendido
- ▶ Parâmetros são específicos para cada algoritmo
 - ▶ **Atenção: alguns parâmetros são específicos da implementação utilizada!**
- ▶ Por exemplo
 - ▶ métodos baseados em informação possuem parâmetros relativos ao número de divisões possíveis (profundidade), relativos ao número mínimo de *datapoints* em cada divisão, relativos a divisão de *features* contínuas, ...
 - ▶ métodos baseados em similaridade possuem parâmetros relativos ao número de vizinhos mais próximos antes de tomar uma decisão, relativos à medida de distância, ...
 - ▶ algoritmos baseados em erros possuem a taxa de aprendizado, entre inúmeros outros que variam de algoritmo a algoritmo

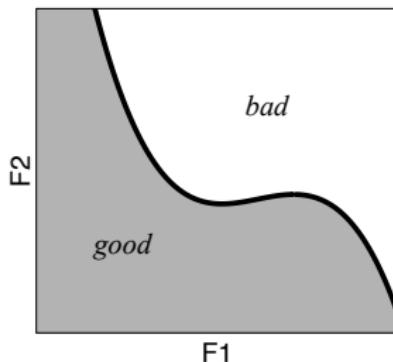
- ▶ O processo de encontrar o melhor conjunto de parâmetros chama-se *fine-tunning*
- ▶ A forma mais comum de se fazer essa busca é um método iterativo chamado *grid search*
 - ▶ para cada parâmetro do algoritmo, define-se uma lista (*força bruta*) ou intervalo de possíveis valores (*busca aleatória*)
 - ▶ o processo de *grid search* treina um modelo usando o *dataset de treino* e testa em um *dataset de validação* mantendo um registro das métricas
 - ▶ ao final do processo, o modelo que possui as melhores métricas é selecionado e avaliado usando-se o *dataset de teste*
 - ▶ a métrica obtida no *dataset de teste* junto com o conjunto de parâmetros e o modelo sem si são retornados como saída
- ▶ *grid search* funciona muito bem em conjunto com *k-fold*
 - ▶ no entanto, um processo de *grid-search* pode demorar minutos, horas ou dias!

- ▶ Frequentemente, nos deparamos com situações que apresentam centenas de *features*
- ▶ Nem sempre todas as features são úteis
 - ▶ Na verdade, features demais fazem com que o modelo entre em *overfitting*

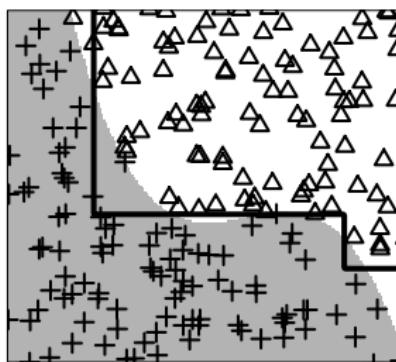


- ▶ Solução: *seleção de features* (ou *feature selection*)
 - ▶ Geração de um subconjunto
 - ▶ Seleção de um subconjunto
 - ▶ Critério de parada
- ▶ Principais métodos:
 - ▶ Seleção sequencial $\rightarrow [1, \dots, N]$
 - ▶ Seleção sequencial reversa $\rightarrow [N, \dots, 1]$
 - ▶ Seleção baseada em teste estatístico $\rightarrow p\text{-value}, \chi^2, \dots$
 - ▶ Seleção baseada em *information gain*
 - ▶ observação: utiliza-se apenas a primeira partição dos *datapoints*
- ▶ *Feature Selection + Grid-search + K-fold*

Resumo

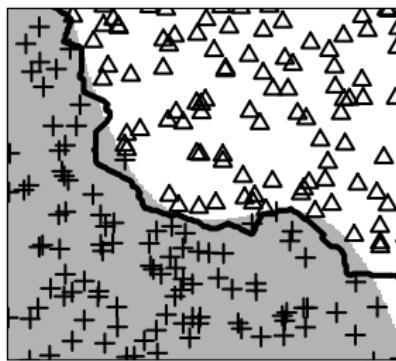


Fonte: John D. Kelleher, Brian Mac Namee, and Aoife D'Arcy. 2015. *Fundamentals of Machine Learning for Predictive Data Analytics: Algorithms, Work Examples and Case Studies*. MIT Press.
<http://machinelearningbook.com/>



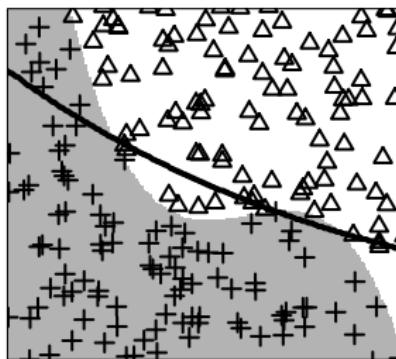
Informação

Fonte: John D. Kelleher, Brian Mac Namee, and Aoife D'Arcy. 2015. *Fundamentals of Machine Learning for Predictive Data Analytics: Algorithms, Work Examples and Case Studies*. MIT Press.
<http://machinelearningbook.com/>



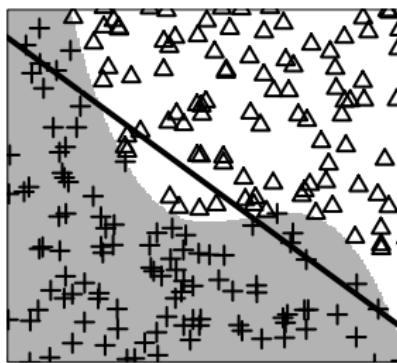
Similaridade

Fonte: John D. Kelleher, Brian Mac Namee, and Aoife D'Arcy. 2015. *Fundamentals of Machine Learning for Predictive Data Analytics: Algorithms, Work Examples and Case Studies*. MIT Press.
<http://machinelearningbook.com/>



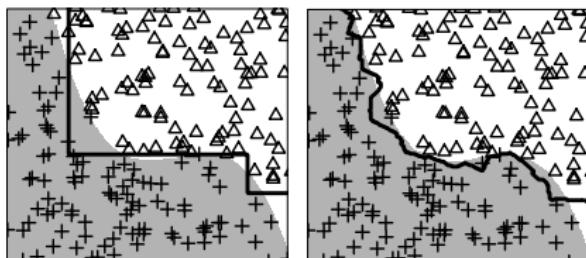
Probabilidade

Fonte: John D. Kelleher, Brian Mac Namee, and Aoife D'Arcy. 2015. *Fundamentals of Machine Learning for Predictive Data Analytics: Algorithms, Work Examples and Case Studies*. MIT Press.
<http://machinelearningbook.com/>



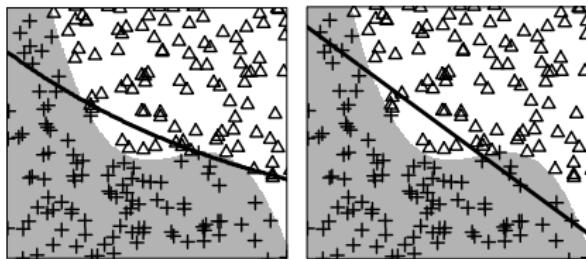
Erro

Fonte: John D. Kelleher, Brian Mac Namee, and Aoife D'Arcy. 2015. *Fundamentals of Machine Learning for Predictive Data Analytics: Algorithms, Work Examples and Case Studies*. MIT Press.
<http://machinelearningbook.com/>



(a) Informação

(b) Similaridade



(c) Probabilidade

(d) Erro

Fonte: John D. Kelleher, Brian Mac Namee, and Aoife D'Arcy. 2015. *Fundamentals of Machine Learning for Predictive Data Analytics: Algorithms, Work Examples and Case Studies*. MIT Press.
<http://machinelearningbook.com/>

- ▶ Não há uma solução simples e fácil para todos os problemas
 - ▶ não há como prever o comportamento de um tipo de algoritmo
 - ▶ os algoritmos mais indicados podem ter a pior performance!
- ▶ A maior parcela do tempo de um projeto de *machine learning* é gasta na parte de entendimento do domínio/problema a ser resolvido e na *Engenharia de features*

Laboratório

Introdução ao Machine Learning com Scikit Learn