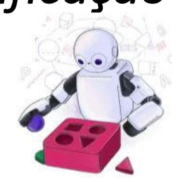
Slide 1

TP555 - Inteligência Artificial e Machine Learning: *Classificação Linear*



Inatel

Felipe Augusto Pereira de Figueiredo
felipe.figueiredo@inatel.br

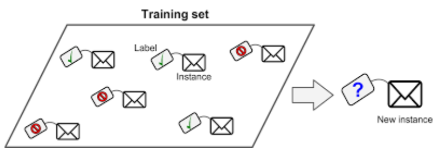
Slide 2

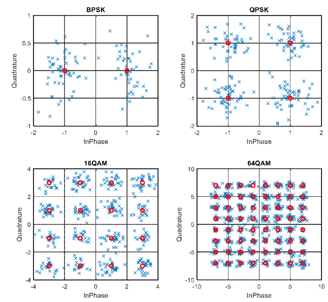
Tópicos abordados

- Abordagens para classificação linear:
  - Classificação Bayesiana (Aula de hoje)
  - Regressão logística (Próxima aula)
- Métricas para avaliação de classificadores (Próxima aula).

Slide 3

Motivação



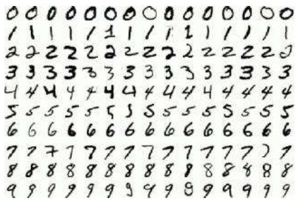


- Classificação de emails entre SPAM e pessoal (HAM).
- Detecção de símbolos (classificação de símbolos).

Motivacão: classificacão de e-mails, detecção de símbolos

Slide 4

Motivação



- Reconhecimento de dígitos escritos à mão.
- Classificação de texto.

Slide 5

Definição do problema

Problema: atribuir a cada exemplo de entrada o ***rótulo*** correspondente a uma das Q classes existentes, $C_q, q = 1, \ldots Q$, à qual o exemplo pertence.

- o Este tipo de desafio é característico de problemas conhecidos como ***classificação***.

- Semelhante ao problema da regressão linear, existe um conjunto de treinamento $\{\boldsymbol{x}(i); y(i)\}_{i=0}^{N-1}$ que é utilizado para treinar um ***classificador***, onde
  - $\boldsymbol{x}(i) = [x_1(i) \quad \cdots \quad x_K(i)]^T \in \mathbb{R}^{K \times 1}$ representa o *i*-ésimo vetor exemplo de entrada, o qual é caraterizado por K atributos, $x_1, \ldots, x_K$
  - e $y(i) \in \mathbb{R}$ representa o *i*-ésimo ***rótulo***. Como veremos à seguir, y pode ser um escalar $\mathbb{R}^1$ ou um vetor $\mathbb{R}^{Q \times 1}$.

Slide 6

Representação da saída desejada

- A saída desejada para o exemplo de entrada deve ser o ***rótulo*** da classe à qual ele pertence.
- Sendo assim, a saída y de um ***classificador***, é uma variável categórica (ou seja, discreta).
- Portanto, para realizarmos o treinamento do modelo, é necessário escolher uma ***representação numérica*** para a saída desejada, ou seja, y.
- Assim, como veremos a seguir, duas opções podem ser adotadas, dependendo do tipo de classificação a ser feita.

Slide 7

Representação da saída desejada

- **Classificação binária**: existem apenas duas classes possíveis, C_1 e C_2. Portanto, neste caso, podemos utilizar ***uma única saída escalar binária*** para indicar a classe correspondente ao exemplo de entrada:

$$y(i) = \begin{cases} 0, & \boldsymbol{x}(i) \in C_1 \\ 1, & \boldsymbol{x}(i) \in C_2 \end{cases}$$

- Assim, $y(i) \in \mathbb{R}^1$, de maneira que o classificador realiza um mapeamento $\mathbb{R}^K \rightarrow \mathbb{R}^1$
- Também é possível utilizar $y(i) = -1$ para $\boldsymbol{x}(i) \in C_1$.

Slide 8

Representação da saída desejada

- **Classificação multi-classes**: existem mais de 2 classes possíveis ($Q > 2$).
- Uma estratégia bastante utilizada para representar estas classes é conhecida como ***one-hot encoding***.
- ***one-hot encoding***: utiliza uma representação binária para cada uma das variáveis categóricas.
- Neste caso, o ***classificador*** produz múltiplas saídas, cada uma representando a ***possibilidade*** (ou probabilidade) do padrão pertencer a uma classe específica.
- **Exemplo**: imaginemos um classificador de texto com quatro classes possíveis: *esporte*, *política*, *ciências* e *variedades*. Como vocês as representariam com o ***one-hot encoding***?

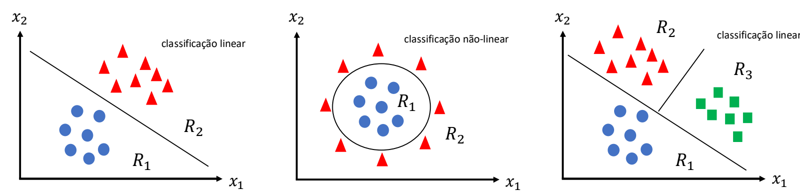
$$\left.\begin{array}{ll} \textit{esporte:} & [1 \quad 0 \quad 0 \quad 0]^T \\ \textit{política:} & [0 \quad 1 \quad 0 \quad 0]^T \\ \textit{ciências:} & [0 \quad 0 \quad 1 \quad 0]^T \\ \textit{variedades:} & [0 \quad 0 \quad 0 \quad 1]^T \end{array}\right\}$$

Assim, $\boldsymbol{y}(i) \in \mathbb{R}^{Q \times 1}$, de maneira que o classificador realiza um mapeamento $\mathbb{R}^K \to \mathbb{R}^Q$.

Slide 9

Fronteiras de decisão de um classificador

- O espaço de entrada $\mathbb{R}^K$ é dividido em ***regiões de decisão*** $R_i, i = 1, \ldots, \mathrm{Q}$, as quais são delimitadas ou separadas pelas ***fonteiras de decisão***, que correspondem a ***superfícies*** (ou ***superfícies de decisão***) no espaço dos atributos onde ocorre uma indeterminação, ou, analogamente, um empate entre diferentes classes possíveis.
- As ***fronteiras de decisão*** podem ser ***lineares*** (e.g., retas e planos) ou ***não-lineares*** (e.g., círculos e esferas).



Regiões de decisão em problemas de classificação binária e multi-classes.

Classificação linear

- Como vimos anteriormente, o objetivo da ***classificação*** é usar as características (i.e., atributos) de, por exemplo, um objeto para identificar a qual classe (ou grupo) ele pertence.
- Um ***classificador linear*** atinge esse objetivo tomando uma decisão de classificação com base no valor de uma ***combinação linear*** dos atributos.
- A saída de um classificador linear é dada por

$$y = f\left(\sum_{k=1}^{K} a_k x_k\right),$$

onde $f(.)$ é uma função que converte o produto escalar dos dois vetores na saída desejada, ou seja, na classe do objeto.

- ***Classificadores lineares*** sãos frequentemente usados em situações em que a velocidade da classificação é um problema, pois ele geralmente é o classificador mais rápido.
- Além disso, os ***classificadores lineares*** geralmente funcionam muito bem quando o número de atributos é grande, como no caso da classificação de documentos.

Teoria bayesiana de decisão

- A teoria bayesiana de decisão é uma ***abordagem estatística*** para o problema de ***classificação***.
- Ela explora o conhecimento de probabilidades ligadas às classes e aos atributos, bem como dos ***custos*** associados a cada decisão, para realizar a classificação de cada novo exemplo.
- **Definições**: Considere que um exemplo (conjunto de atributos) a ser classificado seja descrito por um vetor de atributos $\boldsymbol{x} \in \mathbb{R}^{K \times 1}$. Cada exemplo pertence a uma, e somente uma, classe C_q, sendo que existem ao todo Q classes possíveis.
  - $P(C_q)$ denota a probabilidade ***a priori*** associada à classe C_q.
  - Em outras palavras, $P(C_q)$ indica a probabilidade de um exemplo arbitrário (e desconhecido) pertencer à classe C_q.

Slide 12

Teoria Bayesiana de decisão

- Agora suponha, que um exemplo $\boldsymbol{x}$ seja observado. De posse do conhecimento das características deste exemplo, qual deve ser a decisão quanto à classe a que ele pertence?
  - Uma opção intuitiva e bastante poderosa é escolher a classe que se mostre a mais provável tendo em vista os atributos específicos do exemplo, $\boldsymbol{x}$.
  - Ou seja, a decisão é tomada em favor da classe cuja probabilidade ***a posteriori*** (ou seja, já levando em consideração o conhecimento do vetor de atributos) seja máxima.
  - A probabilidade ***a posteriori*** corresponde à probabilidade condicional $P(C_q|\boldsymbol{x})$.
  - Como calculamos $P(C_q|\boldsymbol{x})$?
- **Teorema de Bayes**

$$P(C_q|\boldsymbol{x}) = \frac{P(\boldsymbol{x}|C_q)P(C_q)}{P(\boldsymbol{x})}$$

onde o termo $P(\boldsymbol{x}|C_q)$ é denominado de ***verossimilhança*** (***likelihood***) e o termo $P(\boldsymbol{x})$ é normalmente chamado de ***evidência***.

Slide 13

Máxima probabilidade a posteriori (MAP)

- A opção intuitiva sugerida anteriormente é conhecida como o critério ou decisor da ***máxima probabilidade a posteriori*** (MAP, do inglês ***maximum a posteriori probability***), cuja decisão para o padrão $\boldsymbol{x}$ é dada pela classe C_q que maximiza $P(C_i|\boldsymbol{x})$, ou seja, em forma matemática:

$$\text{MAP: } C_q = \arg \max_{C_i, i=1,\ldots,Q} P(C_i|\boldsymbol{x}).$$

Probabilidade a posteriori

- Observe que, com base no teorema de Bayes, a solução para a equação acima é equivalente àquela que maximiza o numerador, $P(\boldsymbol{x}|C_i)P(C_i)$, de forma que:

$$\text{MAP: } C_q = \arg \max_{C_i, i=1,\ldots,Q} P(\boldsymbol{x}|C_i)P(C_i),$$

já que o denominador $P(\boldsymbol{x})$ não depende das classes testadas, servindo apenas como fator de escala no critério.

O termo $P(C_i|\boldsymbol{x})$ é chamado de ***probabilidade a posteriori***

Slide 14

Máxima verossimilhança (ML)

- O ***decisor de máxima verossimilhança*** (ML, do inglês ***maximum likelihood***) parte do pressuposto de que não há informação estatística consistente sobre as classes, i.e., sobre $P(C_i)$.
- Portanto, o critério ML toma a decisão em favor da classe que apresenta o maior valor para a probabilidade $P(\boldsymbol{x}|C_i)$. Neste sentido, o ML escolhe a classe C_q mais plausível ou mais verossímil em relação ao padrão observado:

$$\text{ML: } C_q = \arg \max_{C_i, i=1,\ldots,Q} P(\boldsymbol{x}|C_i)$$

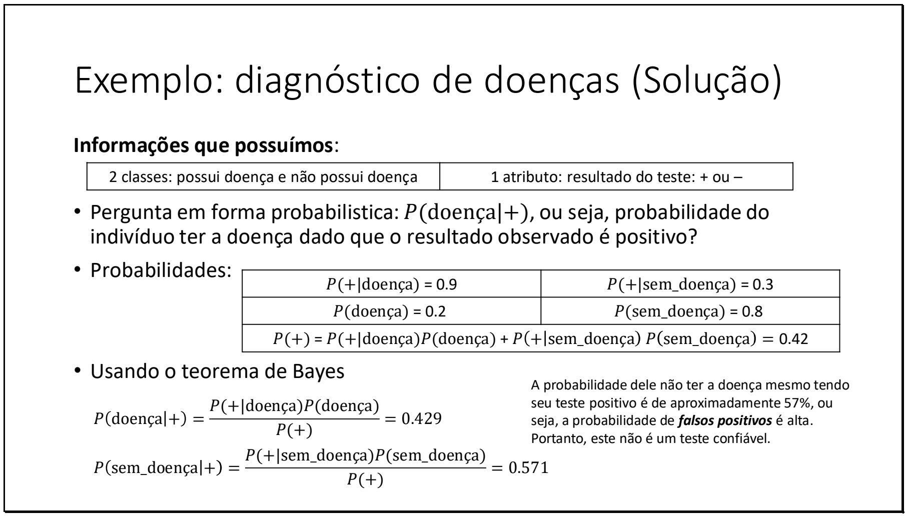
- **OBS**.: se compararmos as expressões associadas aos critérios MAP e ML, percebemos que a diferença fundamental entre eles reside no fato de o MAP explicitamente incorporar o conhecimento das ***probabilidades a priori***, i.e., $P(C_i)$. Curiosamente, quando temos um cenário em que as classes são equiprováveis, i.e., $P(C_i) = 1/Q$ e independentes do índice, i. Então, maximizar a ***probabilidade a posteriori*** fornecerá a mesma solução que o ML.

Exemplo: diagnóstico de doenças

Vamos supor que estamos trabalhando no diagnóstico de uma nova doença, e que fizemos testes em 100 indivíduos distintos. Após coletarmos os resultados, descobrimos que 20 deles possuíam a doença (20%) e 80 estavam saudáveis (80%), sendo que dos indivíduos que possuíam a doença, 90% receberam positivo no teste da doença, e 30% deles que não possuíam a doença também receberam o teste positivo.

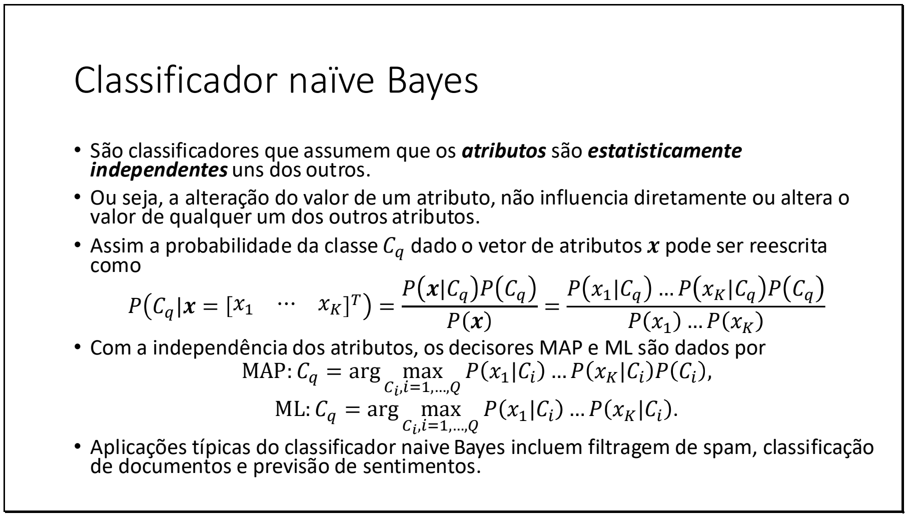
- **Pergunta**: Se uma novo indivíduo realizar o teste e receber um resultado positivo, qual a probabilidade de ele realmente possuir a doença?

Slide 16



Podemos concluir que se o resultado do teste do indivíduo for positivo, ele possui aproximadamente 43% (0.429) de chance de ter a doença e que a chance dele não ter a doença mesmo tendo um teste positivo é de aproximadamente 57% (0.571).

Slide 17



O nome naive (ingênuo em português) é usado porque assume que os atributos do modelo são independentes uns dos outros. Ou seja, a alteração do valor de um atributo, não influencia diretamente ou altera o valor de qualquer um dos outros atributos usados no algoritmo.

A independência dos atributos geralmente não é o caso de problemas do mundo real, onde os atributos podem ter relacionamentos complexos.

Teoria é baseada nos trabalhos do Reverendo Thomas Bayes (1702 a 1761).

Embora a independencia dos atributos possa parecer uma suposição excessivamente simplista (ingênua) em relação aos dados, na prática, o classificador naive Bayes é competitivo com técnicas mais sofisticadas e possui suporte teórico para sua eficácia [1].

As aplicações típicas incluem filtragem de spam [2], classificação de documentos, previsão de sentimentos [3], etc.

Classificação de textos/Filtragem de spam/Análise de sentimento: classificadores Naive Bayes utilizados principalmente em classificação de textos (devido a um melhor resultado em problemas de classes múltiplas e regra de independência) têm maior taxa de sucesso em comparação com outros algoritmos. Como resultado, é amplamente utilizado na filtragem de spam (identificar spam) e Análise de Sentimento (em análise de mídia social, para identificar sentimentos positivos e negativos dos clientes, identificar se o usuário está feliz ou triste ao publicar determinado texto)

References:

[1] Zhang, H. (2004). The Optimality of Naive Bayes. *American Association for Artificial Intelligence*, 1-6.
[2] Samahi, M. (1998). A Bayesian approach to filtering junk e-mail. *AAAI'98 Workshop on Learning for Text Categorization*, 1-8.
[3] Choudhary, V. (2013). Vocal Emotion Recognition Using Naive Bayes Classifier. *Procession of International Conference on Advances in Computer Science, AETACS*, 378-382.

Slide 18

Classificador naïve Bayes

Vantagens

- Fácil de ser implementado e altamente escalável.
- Funciona bem mesmo com poucos dados.
- Rápido para realizar as classificações, e portanto, pode ser utilizado em aplicações de tempo-real.
- Além de simples, ele é conhecido por apresentar performance melhor do que métodos de classificação altamente sofisticados em algumas aplicações.

Desvantagens

- Assume que todos os atributos são independentes, o que muitas vezes não é verdade na prática.
- Não consegue classificar caso uma das probabilidades condicionais seja igual a zero, mas existem formas de se driblar esse problema (e.g., técnica da suavização de Laplace).
- É necessário se conhecer ou se assumir as probabilidades condicionais dos atributos.

Devido à suposição de independência, os classificadores naive Bayes são altamente escaláveis e podem aprender rapidamente a usar atributos de alta dimensão com dados limitados de treinamento. Isso é útil para muitos conjuntos de dados do mundo real, onde a quantidade de dados é pequena em comparação com o número de atributos.

Como determinar a classe mais provável para um exemplo/amostra, **x** = [x1,x2,...,xK]^T, consiste em calcular o produto de K + 1 fatores Q vezes, a notação big-O para a complexidade da classificação em tempo de execução é O(KQ). Isso é computacionalmente muito eficiente e garante ao classificaro naive Bayes sua alta escalabilidade, uma vez que o tempo de execução é escalado linearmente no número de atributos K e no número de classes Q. Isso é especialmente útil com dados apresentando altas dimensões (ou seja, K é grande), como classificação de imagens de alta resolução, como, por exemplo, imagens de ressonância magnética [1].

Na prática, as probabilidades condicionais dos atributos de uma classe são geralmente modeladas usando-se o mesmo tipo de distribuição de probabilidade, como a distribuição binomial ou a distribuição Gaussiana.

Apesar de suas suposições aparentemente simplificadas, os classificadores naive de Bayes têm funcionado muito bem em muitas situações/aplicação do mundo real, sendo famosa sua aplicação em classificação de documentos e filtragem de spam. Eles exigem uma pequena quantidade de dados de treinamento para estimar os parâmetros necessários. (Para as razões teóricas pelas quais classificadores naive Bayes funcionam bem e em quais tipos de dados, consulte a referência [2] abaixo.)

Referencias:

[1] Al-Aidaroos, K. (2012). Medical Data Classification with Naive Bayes Approach. *Information Technology Journal*, *11*, 1166-1174.
[2] H. Zhang (2004). The optimality of Naive Bayes. Proc. FLAIRS.

Tipos de classificadores naïve Bayes

- Na prática, as probabilidades condicionais dos atributos x_k de uma classe, C_q, $P\left(x_k|C_q\right), \forall k$, são geralmente modeladas usando-se o mesmo tipo de distribuição de probabilidade, como as distribuições Gaussiana, Multinominal e de Bernoulli.
- Portanto, tem-se 3 tipos diferentes de classificadores dependendo da suposição feita para a probabilidade condicional $P\left(x_k|C_q\right)$:
  - Classificador naïve Bayes Gaussiano
  - Classificador naïve Bayes Multinomial
  - Classificador naïve Bayes Bernoulli

Slide 20

Classificador naïve Bayes Gaussiano

- Quando lidamos com ***atributos*** $x_1, \ldots, x_K$, que apresentam ***valores contínuos***, uma suposição típica é que os valores dos atributos sejam distribuídos de acordo com uma ***distribuição normal*** (ou **Gaussiana**).
- Para se encontrar os parâmetros do classificador faz-se o seguinte:
  - Primeiro, segmenta-se os atributos, $x_1, \ldots, x_K$, de acordo com a classe a que pertencem;
  - Em seguida, calcula-se a média, μ_{x_k,C_q}, e a variância, $\sigma^2_{x_k,C_q}$, de cada atributo x_k em relação à classe, C_q, a que pertence.
- Assim, a probabilidade condicional $P(x_k|C_q)$ pode ser calculada inserindo-se o valor de x_k na equação da distribuição Normal parametrizada com μ_{x_k,C_q} e $\sigma^2_{x_k,C_q}$.

$$P(x_k|C_q) = \frac{1}{\sigma^2_{x_k,C_q}\sqrt{2\pi}} e^{-(x_k-\mu_{x_k,C_q})^2 / 2\sigma^2_{x_k,C_q}}.$$

- Essa é outra suposição forte, pois muitos atributos não seguem uma distribuição normal. Embora isso seja verdade, supondo uma distribuição normal torna os cálculos muito mais fáceis.

Até agora, vimos os cálculos quando os atributos (x1,x2,...xK) são categóricos. Mas como calcular as probabilidades quando o atributo é uma variável contínua?

Se assumirmos que o vetor de atributos x=[x1,...,xK]^T segue uma distribuição específica, é possível utilizar a função de densidade de probabilidade dessa distribuição para calcular a probabilidade condicional.

Se você assumir que os atributos, x1, ..., xK, seguem uma distribuição normal (também conhecida como gaussiana), o que é bastante comum, substituímos a densidade de probabilidade correspondente de uma distribuição normal e a chamamos de classificado naive Bayes Gaussiano. Você precisa apenas da média e variância dos atributos para calcular as probabilidades condicionais.

O classificador naive Bayes Gaussiano assume que os atributos seguem uma distribuição normal. Essa é outra suposição forte, pois muitos atributos não seguem uma distribuição normal. Embora isso seja verdade, supondo uma distribuição normal torna nossos cálculos muito mais fáceis. Portanto, usamos modelos gaussianos quando os atributos podem assumir infinitos valores.

Slide 21

Exemplo: Probabilidade da prática de esportes

Nesse exemplo vamos usar o classficador Naive Bayes Gaussiano para calcular a probabilidade dos jogadores jogarem ou não, com base nas condições climáticas. Baseado nos dados abaixo, qual a probabilidade dos jogadores jogarem se temperatura = 25 °C e humidade = 62%?

Temperatura [°C]	Humidade [%]	Jogar?
29.44	85	Não
26.67	90	Não
28.33	86	Sim
21.11	96	Sim
20.00	80	Sim
18.33	70	Não
17.78	65	Sim
22.22	95	Não
20.56	70	Sim
23.89	80	Sim
23.89	70	Sim
22.22	90	Sim
27.22	75	Sim
21.67	91	Não

Slide 22

Exemplo: Probabilidade da prática de esportes

Primeiro, precisamos calcular a média e variância para cada atributo, ou seja, para temperatura e humidade.

Temperatura [°C]	
E[temp. | jogar=sim]	22.78
std(temp. | jogar=sim)	3.42
E[temp. | jogar=não]	23.67
std(temp. | jogar=não)	4.39

Humidade [%]	
E[hum. | jogar=sim]	79.11
std(hum. | jogar=sim)	10.22
E[hum. | jogar=não]	86.20
std(hum. | jogar=não)	9.73

P(jogar=sim)	9/14
P(jogar=não)	5/14

P(temp.=25 | jogar=sim) = $\frac{1}{3.42\sqrt{2\pi}} e^{-(25-22.78)^2 / 2(3.42)^2}$ = *0.0944*

P(hum.=62 | jogar=sim) = $\frac{1}{10.22\sqrt{2\pi}} e^{-(62-79.11)^2 / 2(10.22)^2}$ = *0.0096*

P(temp.=25 | jogar=não) = $\frac{1}{4.39\sqrt{2\pi}} e^{-(25-23.67)^2 / 2(4.39)^2}$ = *0.0869*

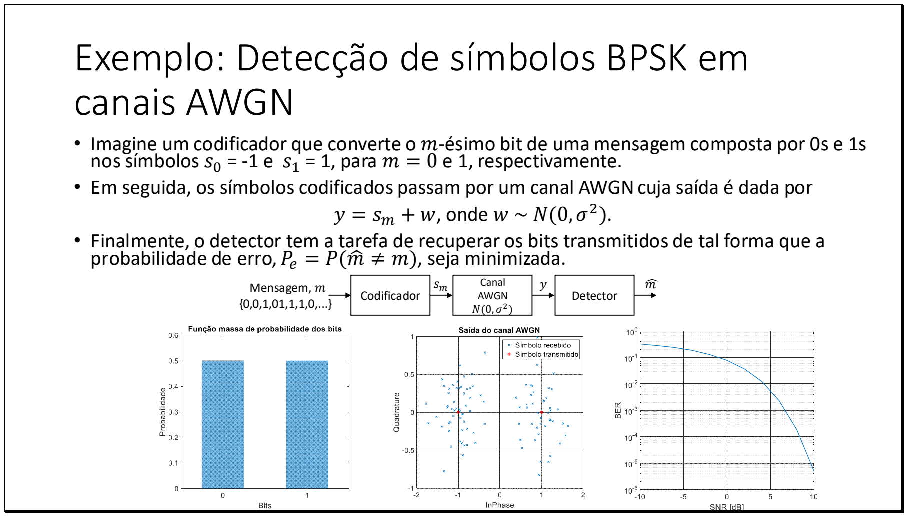
P(hum.=62 | jogar=não) = $\frac{1}{9.73\sqrt{2\pi}} e^{-(62-86.2)^2 / 2(9.73)^2}$ = *0.0019*

Agora calculamos as probabilidades:

- *P*(jogar=sim | temp.=25, hum.=62) = *P*(temp.=25 | jogar=sim) *P*(hum.=62 | jogar=sim)*P*(jogar=sim) = 5.83e-4
- *P*(jogar=não | temp.=25, hum.=62) = *P*(temp.=25 | jogar=não) *P*(hum.=62 | jogar=não)*P*(jogar=não) = 5.78e-5

Portanto, a probabilidade é maior para o caso deles jogarem.

Slide 23



O detector MAP é o detector ideal para modulação BPSK:
http://shannon.cm.nctu.edu.tw/digitalcom/Chap04.pdf
https://blogs.cornell.edu/info2040/2018/11/27/bayes-rule-in-digital-communication-with-bpsk-modulation/

Referências:

[1] https://dsp.stackexchange.com/questions/10222/qpsk-soft-decoding
[2] https://www.tutorialsp
[3] http://www.dsplog.com/2009/09/29/hamming-74-code-with-hard-decision-decoding/oint.com/hard-and-soft-decision-decoding
[4] https://www.gaussianwaves.com/2009/12/hard-and-soft-decision-decoding-2/

Slide 24

Exemplo: Detecção de símbolos BPSK em AWGN

- Detector MAP para esse problema é dado por:

$$S_m = \arg\max_{S_m,\, m=0,1} P(S_m|y) = \arg\max_{S_m, m=0,1} P(y|S_m)P(S_m)$$

- Se o símbolo s_0 é transmitido, então o sinal recebido é dado por: $y = s_0 + w$

$$P(y|s_0) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(y+1)^2 / 2\sigma^2}.$$

- Se o símbolo s_1 é transmitido, então o sinal recebido é dado por: $y = s_1 + w$

$$P(y|s_1) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(y-1)^2 / 2\sigma^2}.$$

- Como $P(S_0)$ = $P(S_1)$ = 1/2, então o detector MAP é equivalente ao ML, e assim

$$S_m = \arg\max_{S_m, m=0,1} \frac{1}{\sigma\sqrt{2\pi}} e^{-(y-S_m)^2 / 2\sigma^2} = \arg\max_{S_m, m=0,1} (y - S_m)^2.$$

O detector MAP é o detector ideal para modulação BPSK:
https://blogs.cornell.edu/info2040/2018/11/27/bayes-rule-in-digital-communication-with-bpsk-modulation/

Referências:

https://ee.stanford.edu/~cioffi/doc/book/chap1.pdf

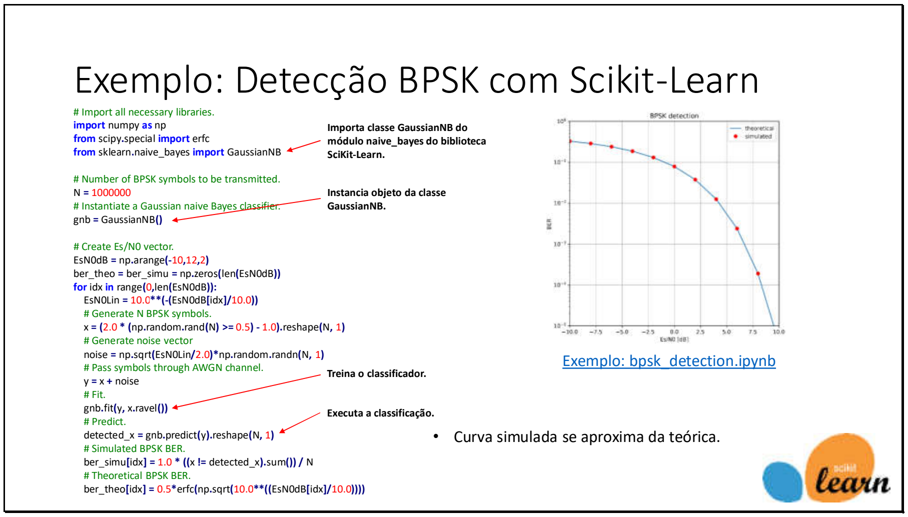
https://dsp.stackexchange.com/questions/10222/qpsk-soft-decoding

https://www.tutorialsp

http://www.dsplog.com/2009/09/29/hamming-74-code-with-hard-decision-decoding/oint.com/hard-and-soft-decision-decoding

https://www.gaussianwaves.com/2009/12/hard-and-soft-decision-decoding-2/

Slide 25



Exemplo: bpsk_detection.ipynb
Link: https://colab.research.google.com/github/zz4fap/tp555-machine-learning/blob/master/exemplos/classification/linear/bpsk_detection.ipynb

Slide 26

Classificador naïve Bayes Multinomial

- Com um classificador naïve Bayes multinomial, os ***atributos*** são ***discretos*** e representam as frequências com as quais determinados eventos são gerados por uma distribuição multinomial, com probabilidades $(p_1, p_2, \dots, p_K)$, onde p_k é a probabilidade de que o evento k ocorra.
- Desta forma, o vetor de atributos $\boldsymbol{x} = [x_1, x_2, \dots, x_K]^T$ é então, um ***histograma***, com x_k contando o número de vezes que o evento k foi observado em uma instância específica.
- Este classificador é normalmente usado para classificação de documentos, com eventos representando a ocorrência de uma palavra no documento.
- A probabilidade de observar um histograma $\boldsymbol{x}$ é dada por

$$P(\boldsymbol{x}|C_q) = \frac{(\sum_k x_k)!}{\prod_k x_k!} \prod_k {p_{qk}}^{x_k},$$

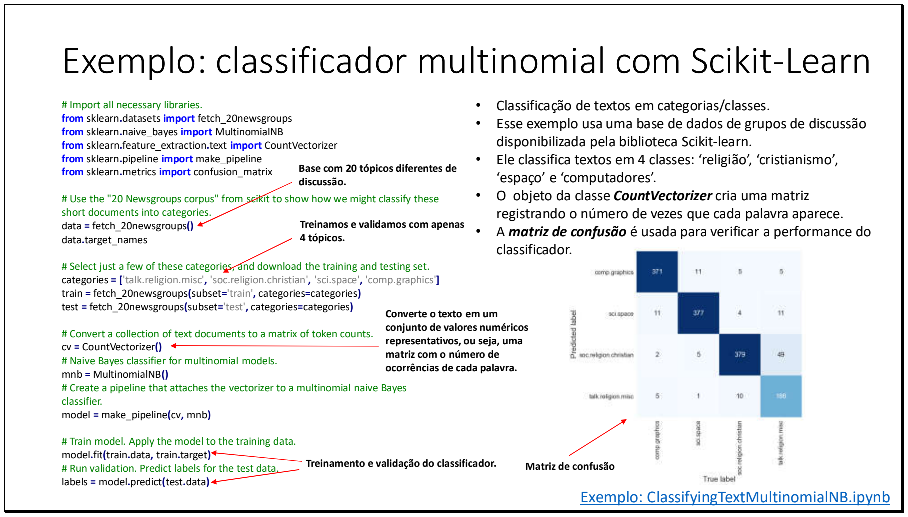
onde p_{qk} é a probabilidade da classe C_q gerar o atributo x_k.

Outro classificador útil é o multinomialNB, onde se supõe que os atributos sejam gerados a partir de uma distribuição multinomial. A distribuição multinomial descreve a probabilidade de observar contagens entre várias categorias e, portanto, multinomialNB é mais apropriado para atributos que representam contagens ou taxas de contagem.

O classificador multinomial Naive Bayes (multinomialNB) é adequado para classificação onde os atributos são discretos (por exemplo, contagem de palavras para classificação de texto).

O classificador multinomialNB se baseia em uma distribuição discreta usada sempre que um atributo deva ser representado por um número inteiro (por exemplo, no processamento de linguagem natural, pode ser a frequência de um termo).

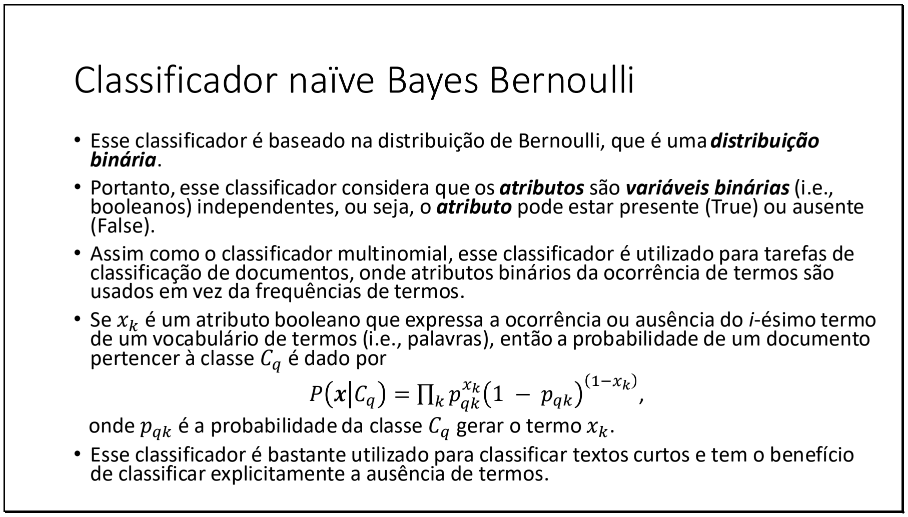
Slide 27



Exemplo: ClassifyingTextMultinomialNB.ipynb
Link: https://colab.research.google.com/github/zz4fap/tp555-machine-learning/blob/master/exemplos/classification/linear/ClassifyingTextMultinomialNB.ipynb
Referência: https://jakevdp.github.io/PythonDataScienceHandbook/05.05-naive-bayes.html

Evidentemente, mesmo esse classificador muito simples pode separar com sucesso a conversa sobre o ***espaço*** da conversa sobre ***computadores***, mas fica um pouco confuso entre conversa sobre ***religião*** e conversa sobre ***cristianismo***. Esta seja talvez uma área esperada de confusão.

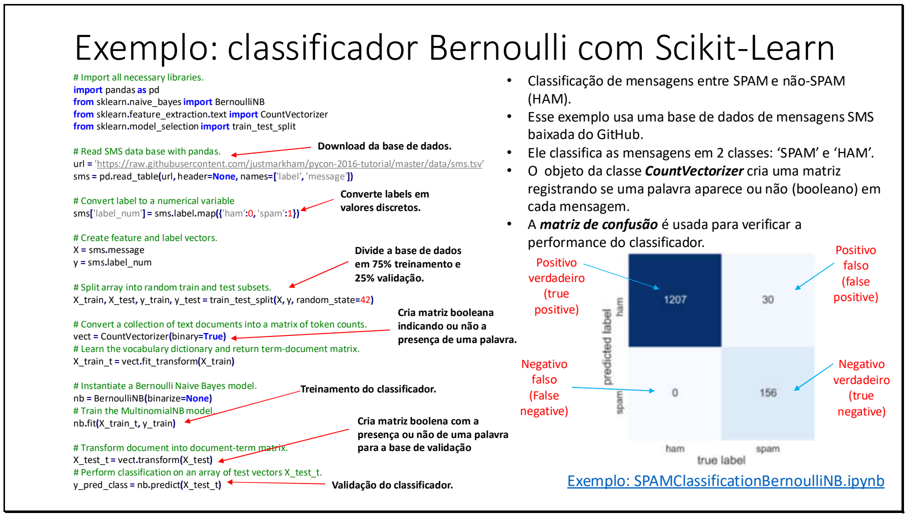
Slide 28



O classificador naïve Bayes Bernoulli (BernoulliNB) é baseado na distribuição de Bernoulli, que é uma uma distribuição binária. BernoulliNB é útil quando um atributo pode estar presente ou ausente. Esse classificador é utilizado quando estamos trabalhando com contagens discretas. Ele conta se um atributo ocorreu ou não.

Assim como o MultinomialNB, esse classificador é adequado para dados discretos. A diferença é que, enquanto o MultinomialNB trabalha com contagens de ocorrências, o BernoulliNB foi projetado para atributos binários/booleanos, ou seja, presenção ou ausnência do atributo.

Slide 29



Exemplo: SPAMClassificationBernoulliNB.ipynb
Link: https://colab.research.google.com/github/zz4fap/tp555-machine-learning/blob/master/exemplos/classification/linear/SPAMClassificationBernoulliNB.ipynb#scrollTo=ARMnYAzud4fo

Slide 30

Tarefas e Avisos

- Na próxima aula veremos regressão logística (classificador linear)
- Exemplos vão estar disponíveis no site.
- Lista #4 vai estar no site ainda hoje e já pode ser resolvida.
- Lista #3 pode ser entregue até dia 21/04.
- **IMPORTANTE**: todos os exercícios que envolvam programação devem estar implementados em um notebook do Jupyter. Eu não terei tempo de rodar todos os repos e através do Notebook eu consigo ver os gráficos e resultados.
- Atendimento às quartas-feiras das 8 as 10 da manhã (Skype: zz4fap)
- https://www.inatel.br/docentes/felipefigueiredo/

Slide 31

Obrigado!