Aprendizagem de Máquina

Modelos Lineares de Classificação

Telmo de Menezes e Silva Filho
tmfilho@gmail.com/telmo@de.ufpb.br
www.de.ufpb.br

UFPB Departamento de ESTATÍSTICA

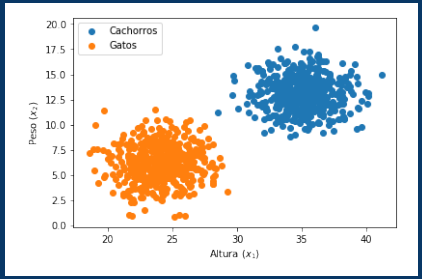
Sumário

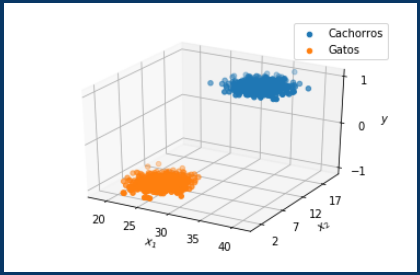
Introdução

- Vamos agora revisitar modelos lineares, mas dessa vez na tarefa de classificação
- Aqui, desejamos encontrar uma função $\hat{f}_k(\mathbf{x}) = \hat{\beta}_{k0} + \hat{\beta}_k^T \mathbf{x}$ que mapeia um vetor de entrada $\mathbf{x}$ a uma de K classes, $k = 1, 2, \ldots, K$
- A **fronteira de decisão** entre duas classes k e l corresponde ao conjunto de pontos em que $\hat{f}_k(\mathbf{x}) = \hat{f}_l(\mathbf{x})$

Classificação usando Regressão Linear

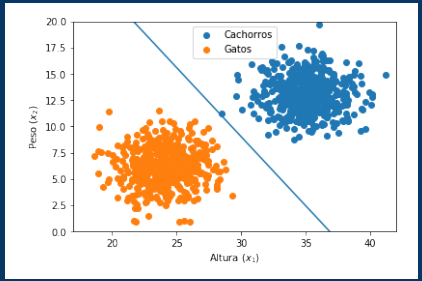
- É possível usar modelos de regressão linear para a tarefa de classificação
- Para isso, escolhemos uma classe positiva e uma classe negativa, e.g. Cachorro $= 1$ e Gato $= -1$

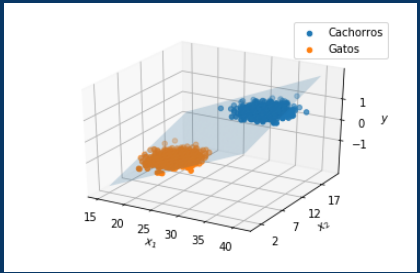




Classificação usando Regressão Linear

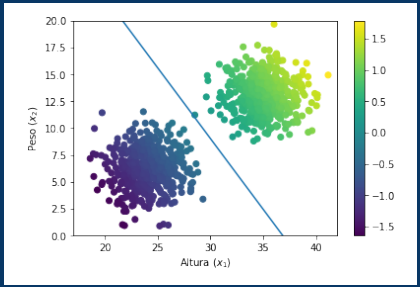
- Assim, podemos ajustar os nossos coeficientes de regressão, encontrando $\hat{y} = -4.21 + 0.11x_1 + 0.09x_2$
- Podemos então classificar novas instâncias como Cachorros, quando $\hat{y} > 0$, ou Gatos, quando $\hat{y} < 0$

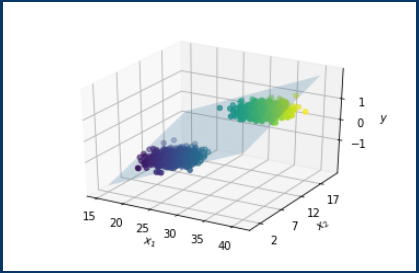




Classificação usando Regressão Linear

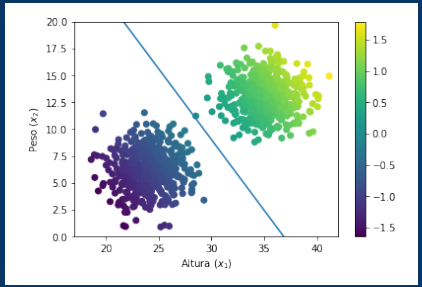
- Assim, podemos ajustar os nossos coeficientes de regressão, encontrando $\hat{y} = -4.21 + 0.11x_1 + 0.09x_2$
- Podemos então classificar novas instâncias como Cachorros, quando $\hat{y} > 0$, ou Gatos, quando $\hat{y} < 0$

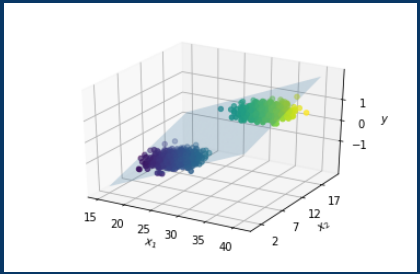




Classificação usando Regressão Linear

- Mas o que fazer quando $\hat{y} = 0$?
- É o que chamamos de **fronteira de decisão**
- Nessa “região”, as duas classes são equiprováveis





Regressão Linear em Problemas Multiclasse

- Como modificamos a regressão linear para lidar com múltiplas classes?
- A resposta é: não mudamos. Podemos usar a regressão linear multivariada
- Para isso, codificamos o vetor de rótulos **y** usando one-hot encoding/variável dummy
- O resultado é uma matriz **Y** de tamanho $N \times K$, com valores $Y_{ik} = 1$, se $y_i = k$, e 0, caso contrário

Regressão Linear em Problemas Multiclasse

- Com a modificação das respostas, podemos agora realizar o ajuste da regressão multivariada

$$\hat{\mathbf{B}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y}$$

- Como vimos na aula anterior, isso equivale a ajustar K modelos de regressão linear simultaneamente
- Dada uma nova observação $\mathbf{x}$ a predição do modelo segue os seguintes passos:
  1. Calcule $\hat{f}(\mathbf{x})^T = (1 : \mathbf{x}^T)\hat{\mathbf{B}}$, um vetor com K dimensões
  2. Identifique o maior componente de $\hat{f}(\mathbf{x})$:

$$\hat{y}_i = \arg\max_k \hat{f}_k(\mathbf{x})$$

Regressão Linear em Problemas Multiclasse

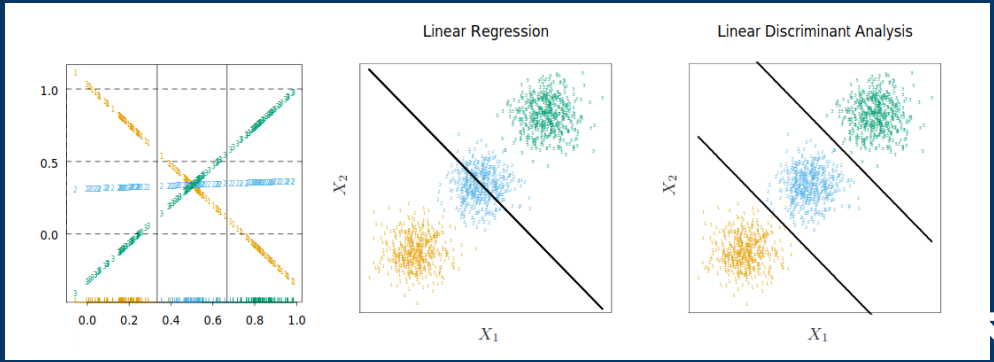
- Devido à forma como a regressão linear multivariada é ajustada, podemos garantir que para todo $\mathbf{x}$

$$\sum_{k=1}^{K} \hat{f}_k(\mathbf{x}) = 1$$

- No entanto, cada $\hat{f}_k(\mathbf{x})$ pode ser negativo ou maior que 1
  - Isso é ainda mais provável de acontecer para instâncias localizadas fora da “caixa” que envolve os dados de treinamento
- Isso significa que as respostas do modelo não são necessariamente boas estimativas de $P(Y = k|\mathbf{x})$

Regressão Linear em Problemas Multiclasse

- Outro possível problema dessa abordagem é o “mascaramento” de algumas classes



Análise de Discriminante

Análise de Discriminante

- Vimos na primeira aula que se soubermos as probabilidades a posteriori das classes $P(Y|X)$ podemos realizar classificações ótimas
  - Classificador **Bayes ótimo**
- Suponha que $f_k(\mathbf{x})$ é a fdp de X dada a classe k e que π_k é a priori de classe, com $\sum_{k=1}^{K} \pi_k = 1$, vimos que podemos aplicar o Teorema de Bayes para obter a posteriori da classe k

$$P(Y = k|X = \mathbf{x}) = \frac{f_k(\mathbf{x})\pi_k}{\sum_{l=1}^{K} f_l(\mathbf{x})\pi_l}$$

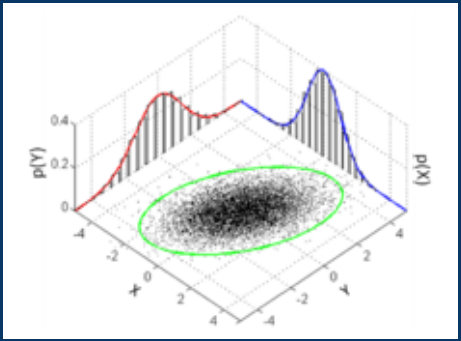
Análise de Discriminante

- Muitas técnicas se baseiam em modelos que consideram as densidades das classes
  - as análises de discriminante linear e quadrático usam densidades Gaussianas
  - Misturas de Gaussianas permitem fronteiras de decisão não-lineares
  - Para mais flexibilidade ainda, temos estimadores de densidade não-paramétricos
  - Naïve Bayes assume que as variáveis são independentes e calcula o produto das densidades marginais (densidades de cada variável)

Análise de Discriminante

- Suponha que a densidade de cada classe seja uma Gaussiana multivariada

$$f_k(\mathbf{x}) = \frac{1}{(2\pi)^{p/2}|\mathbf{\Sigma}_k|^{1/2}} e^{-\frac{1}{2}(\mathbf{x}-\mu_k)^T \mathbf{\Sigma}_k^{-1}(\mathbf{x}-\mu_k)}$$



Análise de Discriminante Linear

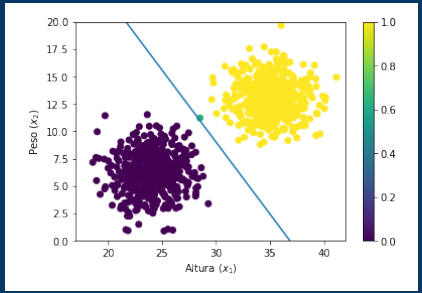
- A análise de discriminante linear (LDA) assume um caso especial em que as classes compartilham a mesma matriz de covariância Σ
- Ao comparar duas classes, podemos observar o log das suas probabilidades a posteriori, chamado de **log-odds**, aí teremos:

$$
\begin{aligned}
log\frac{P(Y=1|X=\mathbf{x})}{P(Y=0|X=\mathbf{x})} &= log\frac{\frac{f_1(\mathbf{x})\pi_1}{f_1(\mathbf{x})\pi_1+f_0(\mathbf{x})\pi_0}}{\frac{f_0(\mathbf{x})\pi_0}{f_1(\mathbf{x})\pi_1+f_0(\mathbf{x})\pi_0}} = log\frac{f_1(\mathbf{x})}{f_0(\mathbf{x})} + log\frac{\pi_1}{\pi_0} \\
&= log\frac{\frac{1}{(2\pi)^{p/2}|\Sigma|^{1/2}}e^{-\frac{1}{2}(\mathbf{x}-\mu_1)^T\Sigma^{-1}(\mathbf{x}-\mu_1)}}{\frac{1}{(2\pi)^{p/2}|\Sigma|^{1/2}}e^{-\frac{1}{2}(\mathbf{x}-\mu_0)^T\Sigma^{-1}(\mathbf{x}-\mu_0)}} + log\frac{\pi_1}{\pi_0} \\
&= log\frac{\pi_1}{\pi_0} - \frac{1}{2}(\mu_1+\mu_0)^T\Sigma^{-1}(\mu_1-\mu_0) \\
&\quad + \mathbf{x}^T\Sigma^{-1}(\mu_1-\mu_0)
\end{aligned}
$$

Análise de Discriminante Linear

- As covariâncias iguais causam o cancelamento dos fatores de normalização e da parte quadrática dos expoentes
- Na prática, não conhecemos as distribuições Gaussianas e precisamos estimá-las usando os dados de treinamento
  - $\hat{\pi}_k = N_k/N$, onde N_k é o número de observações da classe k
  - $\hat{\boldsymbol{\mu}}_k = \sum_{y_i=k} \mathbf{x}_i / N_k$
  - $\hat{\boldsymbol{\Sigma}} = \sum_{k=1}^{K} \sum_{y_i=k} (\mathbf{x}_i - \hat{\boldsymbol{\mu}}_k)(\mathbf{x}_i - \hat{\boldsymbol{\mu}}_k)^T/(N-K)$

LDA no Nosso Exemplo



$$\hat{\pi}_1 = \hat{\pi}_0 = 0.5$$

$$\hat{\mu}_0 = (24.13, 6.14)$$

$$\hat{\mu}_1 = (35.01, 13.06)$$

$$\hat{\Sigma} = \begin{bmatrix} 3.70 & 0.01 \\ 0.01 & 3.09 \end{bmatrix}$$

LDA Multiclasse

- Quando temos $K \geq 3$, podemos escrever a função discriminante de cada classe k como segue:

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

- Aí podemos decidir a classe da nova instância como $\hat{y} = \arg\max_k \delta_k(\mathbf{x})$

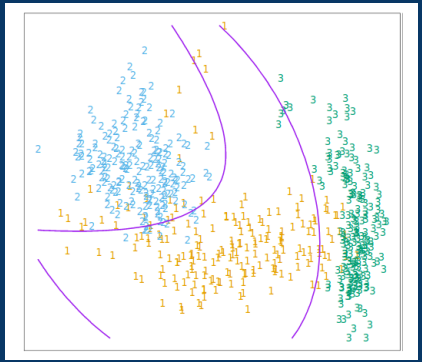
Análise de Discriminante Quadrático

- Se não assumirmos que as matrizes de covariância Σ_k são iguais, os cancelamentos no cálculo dos log-odds não acontecem
- Os fatores quadráticos em **x** permanecem
- Como resultado, temos a análise de discriminante quadrático (QDA)

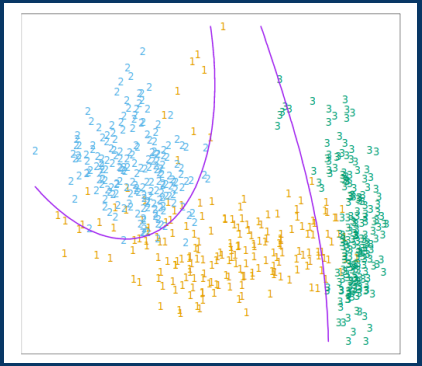
$$\delta_k(\mathbf{x}) = -\frac{1}{2} log|\Sigma_k| - \frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_k)^T \boldsymbol{\Sigma}_k^{-1}(\mathbf{x} - \boldsymbol{\mu}_k) + log\pi_k$$

- Como matrizes de covariâncias separadas tem que ser estimadas para cada classe, se p for grande, isso pode significar um aumento dramático da quantidade de parâmetros do modelo

Análise de Discriminante Quadrático



LDA ajustado em um espaço 5-dimensional $X_1, X_2, X_1X_2, X_1^2, X_2^2$



QDA

Regressão Logística

Regressão Logística

- O modelo de regressão logística é obtido quando modelamos as probabilidades a posteriori de K classes usando funções lineares em $\mathbf{x}$ (incluindo o termo 1 para acomodar o intercepto)

$$
\begin{aligned}
log\frac{P(Y=1|X=\mathbf{x})}{P(Y=K|X=\mathbf{x})} &= \beta_1^T\mathbf{x} \\
log\frac{P(Y=2|X=\mathbf{x})}{P(Y=K|X=\mathbf{x})} &= \beta_2^T\mathbf{x} \\
&\vdots \\
log\frac{P(Y=K-1|X=\mathbf{x})}{P(Y=K|X=\mathbf{x})} &= \beta_{K-1}^T\mathbf{x}
\end{aligned}
$$

Regressão Logística

- A classe escolhida para o denominador não precisa ser a última, mas precisa ser sempre a mesma
- Com isso, podemos calcular as probabilidades a posteriori usando

$$P(Y = k|X = \mathbf{x}) = \frac{e^{\beta_k \mathbf{x}}}{1 + \sum_{l=1}^{K-1} e^{\beta_l \mathbf{x}}}$$

$$P(Y = K|X = \mathbf{x}) = \frac{1}{1 + \sum_{l=1}^{K-1} e^{\beta_l \mathbf{x}}}$$

- Denotamos o conjunto de parâmetros $\theta = \{\beta_1, \beta_2 \ldots, \beta_{K-1}\}$
- E as probabilidades como $P(Y = k|X = \mathbf{x}) = p_k(\mathbf{x}, \theta)$

Regressão Logística

- Com $K = 2$, esse modelo fica super simples, sendo representado por apenas uma função linear, podendo ser escrito como

$$P(Y = 1|X = \mathbf{x}) = \frac{1}{1 + e^{-\beta \mathbf{x}}}$$

- Note que isso equivale a colocar a função $1/(1 + e^{-z})$ sobre um modelo de regressão linear

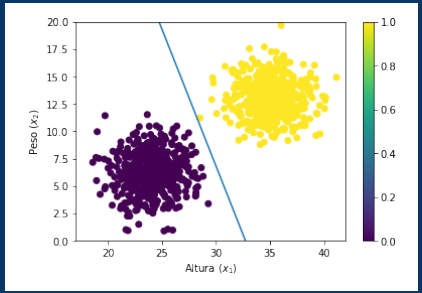
Ajustando a Regressão Logística

- Comumente, ajustamos o modelo usando gradiente descendente ou máxima (log)verossimilhança
- No caso $K = 2$, fazemos $y = 1$ para a classe positiva e $y = 0$ para a classe negativa, além de $p_1(\mathbf{x}, \theta) = p(\mathbf{x}, \theta)$ e $p_2(\mathbf{x}, \theta) = 1 - p(\mathbf{x}, \theta)$
- Assim, queremos maximizar

$$l(\theta) = \sum_{i=1}^{N} \{y_i \log p(\mathbf{x}, \theta) + (1 - y_i) \log(1 - p(\mathbf{x}, \theta))\}$$

- Podemos também minimizar $-l(\theta)$

Regressão Logística no Nosso Exemplo



$$\hat{\beta} = (-1829.71, 55.89, 22.37)^T$$

$$P(Y = 1|X = \mathbf{x}) = \frac{1}{1 + e^{-\hat{\beta}^T \mathbf{x}}}$$

Alguns Outros Detalhes

- Como a regressão logística é basicamente uma regressão linear dos log-odds em função de $\mathbf{x}$, podemos fazer inferência sobre os coeficientes $\hat{\beta}$ encontrados, como vimos na aula anterior
- Além disso, também podemos usar **regularização** L_1 ou L_2 para ajustar o modelo

Perceptron

Perceptron

- Baseado no neurônio artificial de McCulloch–Pitts, Rosenblatt propôs o perceptron, um modelo que encontra um hiperplano que separa as classes, de forma que a distância da fronteira de decisão para as instâncias classificadas incorretamente é mínima
- Revisitaremos o Perceptron, com ajuste ligeiramente diferente, na aula de introdução às redes neurais

Perceptron

- Nesta versão do Perceptron, fazemos $y = 1$ para a classe positiva e $y = -1$ para a negativa
- Para instâncias positivas, queremos que encontrar um hiperplano tal que $\mathbf{x}_i^T \beta > 0$
- A ideia é minimizar a seguinte função de custo (*loss function*)

$$D(\beta) = -\sum_{i \in \mathcal{M}} y_i (\mathbf{x}_i^T \beta)$$

- Onde $\mathcal{M}$ é o conjunto de elementos classificados incorretamente
- Os gradientes dessa função são

$$\partial \frac{D(\beta)}{\partial \beta} = -\sum_{i \in \mathcal{M}} y_i \mathbf{x}_i$$

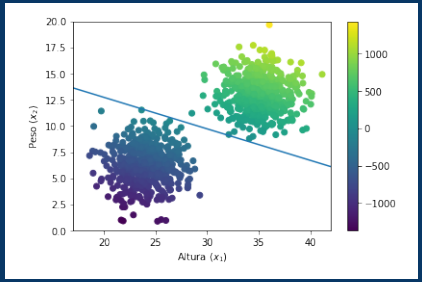
Perceptron

- O algoritmo usa gradiente descendente estocástico para minimizar o critério
- A cada instância apresentada ao algoritmo, ele toma um passo na direção do gradiente negativo (caso ele tenha errado):

$$\boldsymbol{\beta} \leftarrow \boldsymbol{\beta} + \rho y_i \mathbf{x}_i$$

- ρ é a taxa de aprendizagem, que controla o tamanho dos passos tomados na atualização
- O processo se repete até encontrar um hiperplano que separe as classes

Perceptron no Nosso Exemplo

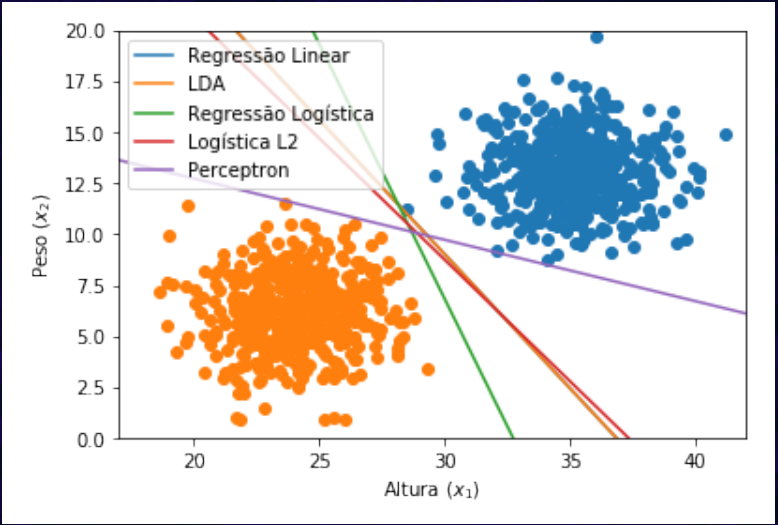


$$\hat{\beta} = (-2289.19, 36.70, 122.08)^T$$

Perceptron

- O perceptron tem algumas limitações
  - Se os dados forem separáveis, existem múltiplas soluções e cada inicialização dos parâmetros levará a uma solução diferente
  - O número de passos para convergir pode ser muito grande (quanto menor a separação das classes, mais passos)
  - Se os dados não forem separáveis, o algoritmo não converge

Todas as Fronteiras de Decisão



Para Terminar

- No final das contas, cada um desses algoritmos encontra um hiperplano que separa as classes
- LDA/QDA e Regressão Logística são definidos de forma que é possível obter probabilidades a posteriori das classes
  - A Regressão Logística modela $P(Y = k|X = \mathbf{x})$ diretamente, enquanto LDA e QDA primeiro assumem Gaussianas para as classes

Para Terminar

- Regressão Linear sempre encontrará o mesmo hiperplano para os mesmos dados, mas o objetivo não é separar os dados o máximo possível, é ajustar os dados da melhor maneira possível
- Perceptron não assume modelo probabilístico algum para os dados e tenta separá-los o máximo que conseguir
  - Mas tem resultados diferentes com inicializações diferentes

Spoilers

- Mais à frente, veremos que existe um algoritmo que tenta encontrar o hiperplano que resulta na separação ótima das classes
- Além disso, veremos transformações que permitem usar métodos lineares mesmo quando a separação entre as classes é não-linear

FIM

Aprendizagem de Máquina

Modelos Lineares de Classificação

Telmo de Menezes e Silva Filho
tmfilho@gmail.com/telmo@de.ufpb.br
www.de.ufpb.br

UFPB Departamento de ESTATÍSTICA