Universidade Federal de São Carlos
Aprendizado de Máquina
Prof. Tiago A. Almeida <talmeida@ufscar.br>

Exercício Prático 3: Regressão Logística

Introdução

Neste exercício você implementará o método de regressão logística e verá como ele utiliza os dados de treinamento para generalizar os dados e encontrar um classificador a fim de fazer predições para amostras não vistas. Antes de começar este exercício, é recomendável que você revise os conceitos apresentados em aula.

Arquivos incluídos neste exercício

`ex03.m` – Script geral do exercício
`ex03Dados1.txt` – Base de dados de treinamento linearmente separável
`ex03Dados2.txt` – Base de dados de treinamento linearmente não separável
`plotarLimiteDecisao.m` – Função para plotar e visualizar o classificador obtido
`visualizarDados.m` – Função para plotar e visualizar os dados da base de treinamento
`atributosPolinomiais.m` – Função para gerar atributos polinomiais a partir dos atributos originais
[$\star$] `sigmoid.m` – Função para calcular o valor da função sigmoidal para uma dada entrada z
[$\star$] `funcaoCusto.m` – Função para calcular a função de custo (J) para um dado θ
[$\star$] `predicao.m` – Função para predizer a classe de uma amostra qualquer
[$\star$] `funcaoCustoReg.m` – Função para calcular a função de custo (J) para um dado θ com regularização

$\star$ indica os arquivos que você precisará completar.

O arquivo `ex03.m` conduzirá todo o processo desse exercício.

O Problema

Você foi contratado por uma grande empresa de cosméticos para desenvolver um método para classificar diferentes espécies de uma flor. Essencialmente, a empresa está interessada em separar automaticamente espécies de uma flor chamada *Iris*. Esse tipo de flor é composta por três espécies: Setosa, Virginica e Versicolour, apresentadas na Figura 1. As duas primeiras (Setosa e Virginica) possuem propriedades aromáticas de interesse da empresa, já a última (Versicolour) não pode ser utilizada.

Devido à forte semelhança visual entre elas, ocorreu a ideia de que, talvez, seja possível detectar cada espécie pelas medidas de comprimento e largura das pétalas. Com base nessa informação, a empresa criou duas base de dados pré-classificadas (Setosa + Versicolour e Virginica + Versicolour) com as respectivas medidas das pétalas das flores. A sua função é implementar o método de regressão logística para determinar a espécie de uma Iris a partir dos dados das pétalas.

(a) Iris Setosa (b) Iris Virginica (c) Iris Versicolour

Figura 1: Espécies de Iris

Visualização dos Dados

Na primeira etapa do procedimento `ex03.m`, a base de dados com exemplos de Iris Setosa e Iris Versicolour é carregada e as amostras são plotadas em 2D para melhor visualização e interpretação dos dados (veja a Figura 2).

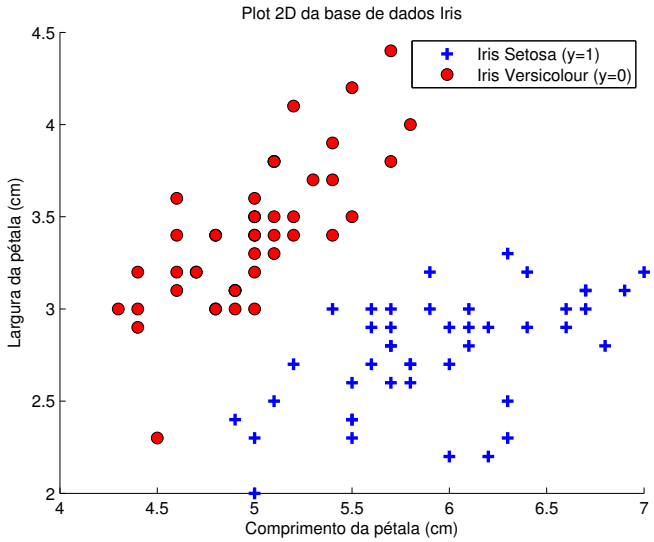

Figura 2: Visualização dos dados

Gradiente Descendente

Nesta parte, você usará o método do gradiente descendente para ajustar os parâmetros da regressão logística (θ) para o conjunto de dados de treinamento.

Equações de ajuste

O objetivo da regressão logística é minimizar a função custo

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}[-y^{(i)}log(h_\theta(x^{(i)})) - (1-y^{(i)})log(1-h_\theta(x^{(i)}))],$$

onde a hipótese $h_\theta(x)$ é determinada pela expressão

$$h_\theta(x) = g(\theta^T x),$$

onde a função g corresponde a função sigmoidal:

$$g(z) = \frac{1}{1+e^{-z}}.$$

*Você precisará completar a função **sigmoid.m**.*

Note que, para um valor positivo grande, o valor da sigmoid correspondente será aproximadamente 1 e para um valor negativo grande, o valor da sigmoid aproximará de 0. Precisamente, para `sigmoid(0)` o valor de retorno deverá ser `0.5`. Além disso, a sua implementação deverá ser capaz de processar vetores e matrizes de entrada. Para uma matriz, a sua função deverá computar a função sigmoidal para cada elemento.

Na rotina `ex03.m`, os dados para a regressão logística já estão previamente configurados.

Função Custo e gradiente

Nesta seção, você precisará implementar a função custo e o gradiente para minimizar os valores de θ.

A sua próxima tarefa é completar o código do arquivo `funcaoCusto.m`. Para isso, lembre-se de que as variáveis X e y não são valores escalares, mas matrizes cujas linhas representam as amostras do conjunto de treinamento.

O gradiente do custo é um vetor de mesma dimensão de θ, sendo que o $j-$ésimo elemento (para $j = 0, 1, \ldots, n$) é definido como:

$$\frac{\partial J(\theta)}{\partial \theta_j} = \frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)})) - y^{(i)})).x_j^{(i)}.$$

*Você precisará completar a função **funcaoCusto.m**.*

Se a sua implementação estiver correta, é esperado que seja exibido um valor de custo para theta inicial aproximadamente igual à **0.693** e gradiente aproximadamente igual à [**0.000; -0.239; 0.161**].

Se você completou os arquivos corretamente, a rotina principal chamará a função `fminunc` para computar o gradiente, otimizar os valor de θ e plotar o classificador resultante.

É esperado que seja plotado um classificador similar ao apresentado na Figura 3. Além disso, o valor da função custo para theta ótimo deverá ser aproximadamente igual à **0.0049**, com $\theta = [-\mathbf{39.365}; \mathbf{15.856}; -\mathbf{14.707}]$.

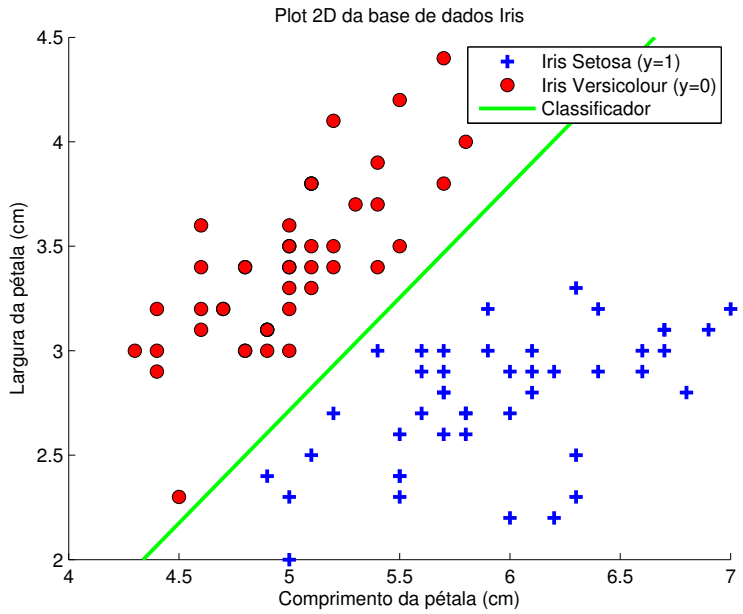

Figura 3: Visualização dos dados e do classificador computado para a base 1

Você precisará completar a função ***predicao.m****.*

Os valores finais obtidos para θ são usados para fazer predições de novos dados. Para uma Iris com pétala de comprimento igual à 5,5cm e largura igual à 3,2cm, é esperado que ela pertença a classe `Setosa` com probabilidade aproximadamente igual à **0.686**.

A seguir, é computada a acurácia do método para a base de treinamento. Se tudo estiver certo, é esperado que a acurácia seja igual à **100**%.

Na última etapa, o programa entra em modo de classificação. É solicitado o comprimento da pétala de uma nova Iris ou `-1` para encerrar a execução.

Regularização

Na segunda etapa do procedimento `ex03.m`, uma base de dados mais complexa com exemplos de Iris Virginica e Iris Versicolour é carregada e as amostras são plotadas (veja a Figura 4). Claramente, as amostras não são linearmente separáveis e, portanto, novos atributos polinomiais precisarão ser criados para melhorar o limite de decisão da regressão logística.

A criação dos atributos polinomiais é realizada pela função `atributosPolinomiais.m` que mapeia o vetor original com apenas duas colunas em um vetor transformado com 28

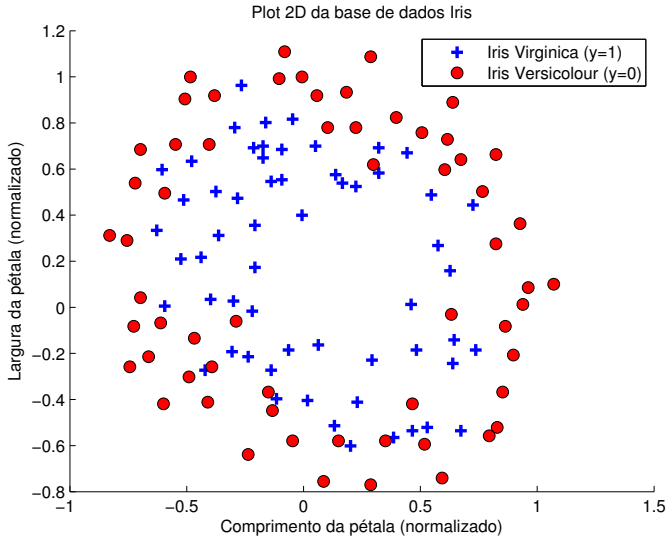

Figura 4: Visualização dos dados

dimensões. Dessa forma, o classificador por regressão logística será treinado com uma base de dimensão mais elevada e produzirá um limite de decisão mais complexo.

Se por um lado o mapeamento dos atributos pode aumentar a robustez do classificador, por outro, ele ficará mais suscetível ao super-ajustamento dos dados (*overfitting*). Assim sendo, para resolver esse impasse você precisará implementar a regressão logística com regularização.

Função Custo e gradiente

A sua próxima tarefa é completar o código do arquivo `funcaoCustoReg.m`. Agora, a equação da regressão logística deverá incorporar a regularização e poderá ser expressa por:

$$J(\theta) = \left[\frac{1}{m}\sum_{i=1}^{m}[-y^{(i)}log(h_\theta(x^{(i)})) - (1-y^{(i)})log(1-h_\theta(x^{(i)}))]\right] + \frac{\lambda}{2m}\sum_{j=2}^{n}\theta_j^2.$$

Note que não é preciso regularizar o parâmetro θ_0. O gradiente da função custo é um vetor no qual o $j-$ésimo elemento é definido como:

$$\frac{\partial J(\theta)}{\partial \theta_0} = \frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)}).x_j^{(i)} \quad (j=0)$$

$$\frac{\partial J(\theta)}{\partial \theta_j} = \left(\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)}).x_j^{(i)}\right) + \frac{\lambda}{m}\theta_j \quad (j \geq 1)$$

Você precisará completar a função ***funcaoCustoReg.m****.*

Se a sua implementação estiver correta, é esperado que seja exibido um valor de custo para theta inicial aproximadamente igual à **0.693**.

A seguir, a rotina principal chamará a função `fminunc` para computar o gradiente, otimizar os valor de θ e plotar o classificador resultante.

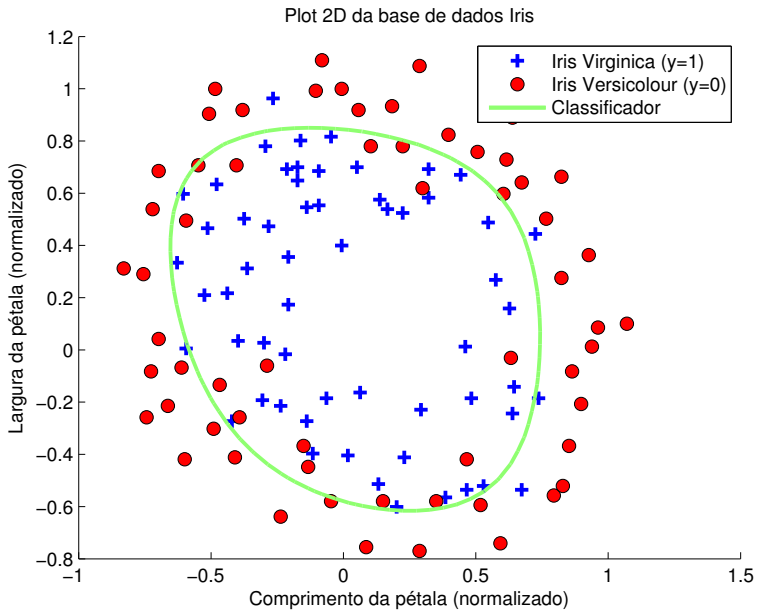

Figura 5: Visualização dos dados e do classificador computado para a base2

É esperado que seja plotado um classificador similar ao apresentado na Figura 5.

Finalmente, é computada a acurácia do método para a base de treinamento. Se tudo estiver certo, é esperado que a acurácia seja igual à **83.05**%.

Na última etapa, o programa entra em modo de classificação. É solicitado o comprimento normalizado da pétala de uma nova Iris ou `-1` para encerrar a execução.