UNIVERSIDADE FEDERAL DO PARANÁ
SETOR DE CIÊNCIAS EXATAS
DEPARTAMENTO DE MATEMÁTICA
CURSO DE MATEMÁTICA

Fillipe Rafael Bianek Pierin

PREVISÃO DE RESULTADOS DE JOGOS DE FUTEBOL POR MEIO DE REGRESSÃO LOGÍSTICA MULTINOMIAL

CURITIBA

2016

Fillipe Rafael Bianek Pierin

PREVISÃO DE RESULTADOS DE JOGOS DE FUTEBOL POR MEIO DE REGRESSÃO LOGÍSTICA MULTINOMIAL

Trabalho realizado na disciplina estágio obrigatório para obtenção do título de Bacharel em Matemática.

Orientador: Geovani Nunes Grapiglia

CURITIBA

2016

Resumo

Este trabalho apresenta um modelo de regressão logística multinomial para previsão de resultados de jogos de futebol. Os parâmetros do modelo foram calibrados a partir de dados do campeonato brasileiro de 2016 retirados do site da Confederação Brasileira de Futebol (CBF) e do site Diário Catarinense (DC). A calibração dos parâmetros foi realizada de modo a se minimizar o erro entre as previsões do modelo e os resultados reais. Para resolver o problema de minimização irrestrito correspondente, utilizou-se a função fminunc do Octave. O modelo obtido apresentou uma taxa de acerto de 55%. Além disso, comparado com modelos usados em sites da internet este modelo obteve resultados satisfatórios. Para a 38º rodada do campeonato brasileiro de futebol de 2016, o modelo teve acerto em (6 de 10 jogos) como o melhor dos cinco sites comparados.

Palavras-chaves: regressão logística multinomial, otimização, futebol, jogos.

Lista de Figuras

Lista de Tabelas

Sumário

1 INTRODUÇÃO

1.1 MOTIVAÇÃO

Os esportes fazem parte da vida de todos, principalmente o futebol que é a paixão nacional no Brasil. Muitas vezes vemos pessoas discutindo qual time é o melhor, qual time tem os melhores jogadores, quem ganhará o campeonato, quem será rebaixado, etc. Isso torna muito interessante o estudo e análise de dados relacionados ao futebol.

1.2 OBJETIVO DO ESTUDO

O objetivo deste trabalho é fazer a implementação e treinamento de um modelo de regressão logística multinomial, com o intuito de fazer a previsão de resultados dos jogos de futebol. Esse tipo de previsão pode ser visto como um problema de classificação, pois queremos classificar os resultados dos jogos de futebol em vitória do time mandante, empate ou vitória do time visitante. Utilizaremos dados do campeonato brasileiro de 2016.

Este trabalho está organizado da seguinte maneira. No Capítulo 2 faz-se uma revisão teórica do modelo de regressão logística multinomial. No Capítulo 3 apresenta-se a aplicação do modelo de regressão logística multinomial à previsão de resultados de jogos de futebol. Por fim, no Capítulo 4 são apresentados as conclusões deste trabalho.

2 REGRESSÃO LOGÍSTICA

O modelo de regressão logística é uma técnica, que por meio de um conjunto de observações, produz-se um modelo para predição de dados (valores categóricos), que é de forma binária (0 ou 1), sucesso ou fracasso de certo evento. Esse modelo serve para a partir do conjunto de dados (variáveis independentes) predizer a variável resposta (variável dependente). A variável dependente é disposta em uma ou mais categorias, diferente da regressão simples em que a variável resposta é contínua e a variável resposta é expressa através de probabilidade de ocorrência. E a variável resposta é um valor numérico, ao contrário do modelo regressão simples.

Para modelar os casos em que a variável resposta é binária, se deve considerar os pressupostos que a variável dependente $y \in \{0, 1\}$, onde 0 é "classe negativa" e 1 é " classe positiva ". Desta forma, o valor esperado (resposta) é uma probabilidade (p), dada pela função logística, que possui valor entre 0 e 1.

Deseja-se modelar um modelo $m_\theta(x)$ tal que $m_\theta(x) = p(y = 1|x; \theta)$, ou seja, probabilidade de $y = 1$ dado que x ocorreu, considerando o parâmetro θ. Para esta finalidade, o modelo de regressão linear $m_\theta(x) = \theta^T x = \theta_0 + \theta_1 x_1 + \cdots + \theta_n x_n$ não é mais adequado, pois podemos ter $m_\theta(x) \notin [0, 1]$.

O modelo de regressão logística consiste em tomar $m_\theta(x) = g(\theta^T x)$ com $g(z) = \dfrac{1}{1 + e^{-z}}$. Tal função g é chamada de função logística e dá o nome a este tipo de regressão.

2.1 FUNÇÃO LOGÍSTICA

A partir do estudo a função logística g, verifica-se que $g(z) = \dfrac{1}{1 + e^{-z}} \in [0, 1]$ para todo $z \in \mathbb{R}$, além disso, $\lim\limits_{z \to +\infty} g(z) = 1$ e $\lim\limits_{z \to -\infty} g(z) = 0$. Também calcula-se e analisa-se as derivadas da função logística:

primeira derivada,

$$g'(z) = \frac{e^z(e^z+1) - e^z(e^z)}{(1+e^z)^2} = \frac{e^z + e^{2z} - e^{2z}}{(e^z+1)^2} = \frac{e^z}{(1+e^z)^2}$$

com $-\infty < x < \infty$

segunda derivada,

$$g''(z) = \frac{e^z(1+e^z)^2 - e^z(1+e^z)e^z}{(1+e^z)^4} = \frac{e^z(1+2e^z+e^{2z}) - 2e^{2z}(1+e^z)}{(1+e^z)^4}$$
$$= \frac{e^z + 2e^{2z} + e^{3z} - 2e^{2z} - 2e^{3z}}{(1+e^z)^4} = \frac{e^z - e^{3z}}{(1+e^z)^4}$$

Com a análise da função de $g(z)$ e seu gráfico, se verifica a partir da segunda derivada, que como tanto a numerador quanto o denominador é positivo, então $g(z)$ é uma função crescente, visto no gráfico 1. Igualando a segunda derivada a zero, obtém-se que o ponto crítico é $(z = 0)$, e que a concavidade é voltada para baixo quando $z > 0$ e para cima de $z < 0$. Veja,

igualando a zero para encontrar o ponto crítico,

$$g''(z) = 0 = \frac{e^z - e^{3z}}{(1+e^z)^4} \Rightarrow e^z = e^{3z} \Rightarrow z = 3z \Rightarrow 2z = 0 \Rightarrow z = 0$$

Portanto, o gráfico de função g tem o formato de "S" (sigmoide) conforme o gráfico 1.

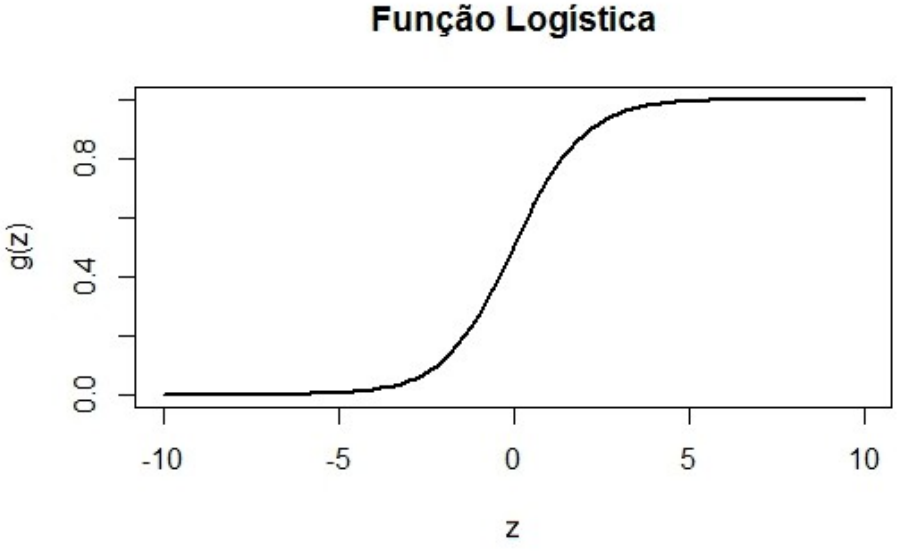

Figura 1: Função Logística

2.2 PREVISÃO DO MODELO

Da seção anterior tem-se que

$$\begin{cases} g(z) \geq 0.5, \text{se } z \geq 0 \\ g(z) < 0.5, \text{se } z \geq 0 \end{cases}.$$

Consequentemente, a previsão usando o modelo logístico pode ser feito da seguinte maneira:

- Prever "$y = 1$" quando $m_\theta(x) = g(\theta^T x) \geq 0.5$, ou seja, quando $\theta^T x \geq 0$;
- Prever "$y = 0$" quando $m_\theta(x) = g(\theta^T x) < 0.5$, ou seja, quando $\theta^T x < 0$.

2.3 ESTIMAÇÃO OS PARÂMETROS

Quando se ajusta um modelo de regressão, deve-se estimar os parâmetros θ_0, θ_1, $\ldots$, θ_n do modelo. Veja como fazer isso de modo que, as respostas do modelo logístico sejam coerentes com o conjunto de dados considerado.

Suponha que temos m pontos $(x^{(1)}, y^{(1)})$, $(x^{(2)}, y^{(2)})$, $\ldots$, $(x^{(m)}, y^{(m)}) \in \Omega$, onde $\Omega = \{(x, y) \in \mathbb{R}^m \text{x} \{0, 1\} \, | x = (x_0, x_1, \ldots, x_n) \text{ com } x_0 = 1\}$.

A medida natural para discrepância entre um modelo $m_\theta(x)$ e o conjunto de dados y é

$$f(\theta) = \sum_{i=1}^{m} (m_\theta(x^{(i)}) - y^{(i)})^2 = \text{custo } (m_\theta(x^{(i)}), y^{(i)}) \tag{2.1}$$

Note que, se $m_\theta(x) = \theta^T x$ (isto é, se considerar o modelo múltiplo), então f é uma função quadrática convexa, quando o conjunto de dados resulta em uma matriz com colunas linearmente dependentes. Porém, quando se considera $m_\theta(x) = g(\theta^T x)$, sendo g a função logística, perde-se a convexidade da função f.

A fim de se obter uma função objetivo convexa (para facilitar a otimização), considera-se a seguinte função custo

$$custo(m_\theta(x^{(i)}), y^{(i)}) = \begin{cases} -log(m_\theta(x)), \text{se } y = 1 \\ -log(1 - m_\theta(x)), \text{se } y = 0 \end{cases} \tag{2.2}$$

Temos duas possibilidades de gráfico dessa função:

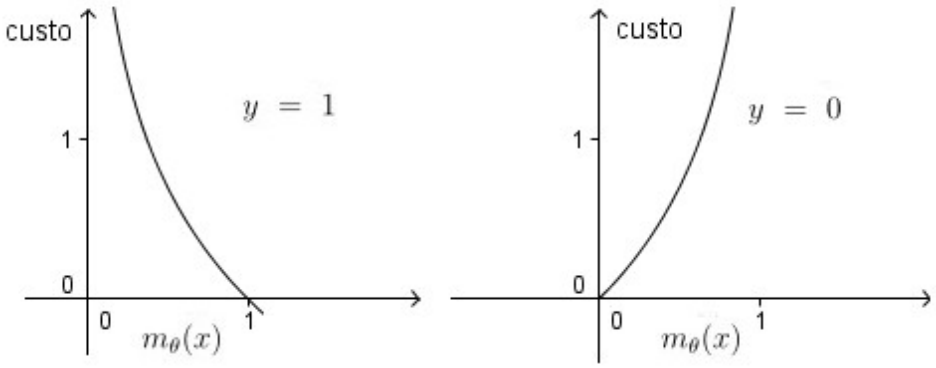

Figura 2: Função custo.

Veja que:

- se $y = 1$ e $m_\theta(x) = 1$, então o custo $(m_\theta(x^{(i)}), y^{(i)}) = 0$. Além disso, custo $(m_\theta(x^{(i)}), y^{(i)}) \to +\infty$, quando $m_\theta(x) \to 0$;
- se $y = 0$ e $m_\theta(x) = 0$, então o custo $(m_\theta(x^{(i)}), y^{(i)}) = 0$. Além disso, custo $(m_\theta(x^{(i)}), y^{(i)}) \to +\infty$, quando $m_\theta(x) \to 1$.

Considerando a função custo dada pela equação 2.2, a estimação do parâmetro θ a partir do conjunto de dados, se obtém pela resolução do seguinte problema de otimização irrestrita

$$\min_\theta \ f(\theta) = \sum_{i=1}^{m} \left[y^{(i)} log(m_\theta(x^{(i)})) + (1 - y^{(i)}) log(1 - m_\theta(x^{(i)})) \right]$$

cuja função objetivo pode-se verificar ser convexa. Do ponto de vista de otimização isso é excelente, pois todo ponto crítico de uma função convexa e diferenciável é um minimizador global da função. Portanto, se obtém uma boa aproximação de um minimizar global de f aplicando-se qualquer um dos métodos usuais de otimização:

- Método do Gradiente;
- Método de Newton;

- Método Quase-Newton;
- Método de Região de Confiança.

Todos os métodos citados acima exigem o gradiente da função objetivo f. Para calcular o gradiente de f, convém reescrever a função custo da seguinte forma

$$f(\theta) = -\frac{1}{m}\sum_{i=1}^{m}[y^{(i)}log(m_\theta(x^{(i)})) + (1-y^{(i)})log(1-m_\theta(x^{(i)}))] \tag{2.3}$$

Agora se deve calcular o gradiente da f (∇f), pois o mesmo é necessário para minimizar f usando os métodos usuais de otimização irrestrita.

Seja $f(\theta) = -\frac{1}{m}\sum_{i=1}^{m}[a+b]$, onde $a = y^{(i)}log(m_\theta(x^{(i)}))$ e $b = (1-y^{(i)})log(1-m_\theta(x^{(i)}))$. Então,

$$\begin{aligned}
\frac{\partial a}{\partial \theta_j} &= y^{(i)}\frac{1}{m_\theta(x^{(i)})}\frac{\partial m_\theta(x^{(i)})}{\partial \theta_j} \\
&= y^{(i)}\frac{1}{m_\theta(x^{(i)})}(m_\theta(x^{(i)}))^2(-e^{-\theta^T x^{(i)}}x_j^{(i)}) \\
&= y^{(i)}\frac{-e^{-\theta^T x^{(i)}}}{1+e^{-\theta^T x^{(i)}}}x_j^{(i)} \\
&= -y^{(i)}x_j^{(i)}\frac{1}{1+e^{\theta^T x^{(i)}}} \\
&= -y^{(i)}x_j^{(i)}m_\theta(-x^{(i)}) \\
&= -y^{(i)}x_j^{(i)}(1-m_\theta(x^{(i)}))
\end{aligned}$$

e

$$\begin{aligned}
\frac{\partial b}{\partial \theta_j} &= (1-y^{(i)})\frac{\partial}{\partial \theta_J}log(1-\frac{1}{1+e^{-\theta^T x^{(i))}}}) \\
&= (1-y^{(i)})\frac{\partial}{\partial \theta_j}log(\frac{e^{-\theta^T x^{(i)}}}{1+e^{-\theta^T x^{(i)}}}) \\
&= (1-y^{(i)})\frac{\partial}{\partial \theta_j}log(\frac{1}{1+e^{\theta^T x^{(i)}}}) \\
&= (1-y^{(i)})\frac{\partial}{\partial \theta_j}log(m_\theta(-x^{(i)})) \\
&= (1-y^{(i)})x_j^{(i)}m_\theta(x^{(i)})
\end{aligned}$$

Logo, o gradiente do custo é um vetor onde a j-ésima entrada é definido como

$$\begin{aligned}
\frac{\partial f(\theta)}{\partial \theta_j} &= -\frac{1}{m}\sum_{i=1}^{m}[-y^{(i)}x_j^{(i)}m_\theta(-x^{(i)}) + (1-y^{(i)})x_j^{(i)}m_\theta(x^{(i)})] \\
&= -\frac{1}{m}\sum_{i=1}^{m}[-y^{(i)}x_j^{(i)}(1-m_\theta(x^{(i)})) + (1-y^{(i)})x_j^{(i)}m_\theta(x^{(i)})] \\
&= -\frac{1}{m}\sum_{i=1}^{m}[-y^{(i)}x_j^{(i)} + y^{(i)}x_j^{(i)}m_\theta(x^{(i)}) + x_j^{(i)}m_\theta(x^{(i)}) - x_j^{(i)}m_\theta(x^{(i)})y^{(i)}] \\
&= -\frac{1}{m}\sum_{i=1}^{m}(m_\theta(x^{(i)}) - y^{(i)})x_j^{(i)} \\
&= \left(\begin{bmatrix} m_\theta(x^{(1)}) \\ \vdots \\ m_\theta(x^{(m)}) \end{bmatrix} = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(m)} \end{bmatrix}\right)^T \begin{bmatrix} X_j^{(1)} \\ \vdots \\ X_j^{(m)} \end{bmatrix}
\end{aligned}$$

Portanto, se obtém que o gradiente da função custo é

$$\begin{aligned}
\nabla f(\theta) &= \left(\sum_{i=1}^{m}(m_\theta(x^{(i)}) - y_i)x_0^{(i)}, \ldots, \sum_{i=1}^{m}(m_\theta(x^{(i)}) - y_i)x_m^{(i)}\right) \\
&= \big((\alpha_\theta - y)^T)A_o, \ldots, (\alpha_\theta - y)^T)A_m\big) \\
&= \big((\alpha_\theta - y))A_o^T, \ldots, (\alpha_\theta - y))A_m^T\big)
\end{aligned}$$

Ou seja,

$$\nabla f(\theta) = X^T(\alpha_\theta - y) \tag{2.4}$$

$$\text{onde } X^T = \begin{bmatrix} A_o^T \\ \vdots \\ A_n^T \end{bmatrix},\ X = \begin{bmatrix} x_0^{(1)} & \ldots & x_0^{(m)} \\ x_1^{(1)} & \ldots & x_1^{(m)} \\ \vdots & \ddots & \vdots \\ x_n^{(1)} & \ldots & x_n^{(m)} \end{bmatrix} = \begin{bmatrix} | & & | \\ x^{(1)} & \ldots & x^{(m)} \\ | & & | \end{bmatrix}.$$

2.3.1 CLASSIFICAÇÃO

Usando a expressão do gradiente da f dada em 2.4, pode-se minimizar f aplicando qualquer um dos métodos usuais de otimização, já citados. Quando calculado o parâmetro θ, dado $x \in \mathbb{R}^{n+1}$ se pode classificá-lo da seguinte forma:

- Prever "$y = 1$" , se $m_\theta(x) \geq 0.5$;

- Prever "$y = 0$" , se $m_\theta(x) < 0.5$.

2.4 REGRESSÃO LOGÍSTICA MULTINOMIAL

O processo visto na seção anterior apenas nos permite fazer classificação binária, isto é, quando se assume a existência de apenas duas classes $y \in \{0, 1\}$. Agora considera-se o problema de classificação quando temos mais de duas classes, como é caso da modelagem de dados de futebol, onde se quer classificar os dados de futebol, em vitória do time mandante, empate ou vitória do time visitante. Para isso, usa-se o método One-vs-All.

2.4.1 MÉTODO ONE-VS-ALL

O método One-vs-All caracteriza-se pelo treino de vários classificadores únicos para cada classe. As amostras pré-classificadas de uma mesma classe são positivas e o restante negativas. Criam-se n divisões que, em cada etapa da aprendizagem uma classe é comparada com as demais classes. A partir da escolha de um classificador, em comparação com as demais classes, n classificadores são criados. O método One-vs-All treina o classificador da regressão logística $m_\theta^{(i)}(x) = p(y = 1|x; \theta)$, para cada classe (i) predizer a probabilidade de $y = j$. Assim, pelo método One-vs-All, se

$$j = argmax_{i \in \{1,2,3\}} \left\{ m_\theta^{(i)}(x) \right\}$$

prevemos que o jogo x terá como resultado j.

O One-vs-All classifica os resultados dos jogos futuros, em vitória do time mandante ($m_\theta^{(1)}(x)$), empate ($m_\theta^{(2)}(x)$) ou vitória do time visitante ($m_\theta^{(3)}(x)$), usando dados das rodadas anteriores a rodada em que se deseja fazer a previsão.

2.4.2 PREVISÃO DE NOVOS DADOS

Para fazer a previsão de novos dados ($\hat{y}$) usa-se o método One-vs-All. Com esse intuito, maximiza-se a classe (i) usando a função logística com os parâmetros estimados anteriormente.

$$\max_i \ m_\theta^{(i)}(x)$$

3 APLICAÇÃO

3.1 INTRODUÇÃO

Segundo (FARIAS, 2008), cada autor ao trabalhar com dados de campeonatos de futebol tem sua interpretação dos parâmetros (θ) do modelo usado. Os parâmetros do modelo podem ou não influenciar na previsão do resultado de uma partida. Desta forma, construiu-se um banco de dados com as variáveis que se suponha ajudar na previsão dos jogos de futebol. Em seguida fez-se as previsões de futuras rodadas usando o modelo de regressão logística multinomial. Após verificou-se a taxa de acerto do modelo usando o método One-vs-All.

Para a implementação e análise dos resultados, utilizou-se como ferramenta computacional o software Octave (EATON DAVID BATEMAN; WEHBRING, 2015).

3.2 BANCO DE DADOS

O banco de dados utilizado neste estudo foi construído a partir de dados do campeonato brasileiro de 2016. Esses dados foram obtidos do site da Confederação Brasileira de Futebol (CBF) `http://www.cbf.com.br/` e do site Diário Catarinense (DC) `http://dc.clicrbs.com.br/sc/esportes`. Neste campeonato temos 20 times, com 380 jogos no total, sendo 10 por rodada. O banco de dados construído possui as seguintes variáveis $(x_i), i \in \{1, 2, \ldots, 18\}$:

- x_1 = colocação do time A antes do jogo;
- x_2 = colocação do time B antes do jogo;
- x_3 = número de vitórias do time A nos últimos dois jogos;
- x_4 = número de vitórias do time B nos últimos dois jogos;

- x_5 = número de empates do time A nos últimos dois jogos;
- x_6 = número de empates do time B nos últimos dois jogos;
- x_7 = saldo de gols do time A antes do jogo;
- x_8 = saldo de gols do time B antes do jogo;
- x_9 = número de cartões amarelos do time A antes do jogo;
- x_{10} = número de cartões amarelos do time B antes do jogo;
- x_{11} = número de cartões vermelhos do time A antes do jogo;
- x_{12} = número de cartões vermelhos do time B antes do jogo;
- x_{13} = número de escanteios do time A antes do jogo;
- x_{14} = número de escanteios do time B antes do jogo;

Ainda neste banco de dados temos a variável resposta (y_i), relativo aos resultados anteriores à rodada em que se deseja fazer a previsão. A variável resposta possui os seguintes valores:

- 1 = vitória do time mandante;
- 2 = empate;
- 3 = vitória do time visitante.

As 14 variáveis foram usadas no modelo, pois se analisou que elas ajudavam a classificar os jogos. E os motivos, em que levou o uso dessas variáveis são as seguintes:

- x_1 e x_2 porque a colocação do time antes da rodada, a ser prevista, mostra o quanto o time tem de desempenho até o momento do campeonato. Desta forma se espera que times em melhor colocação ganhem e com menor colocação percam;
- x_3, x_4, x_5 e x_6 que são números de vitórias e empates nos dois últimos jogos, pela razão que o time que vem embalado no campeonato, tem maior chance de ganhar o próximo jogo ou pelo menos se espera;
- x_7 e x_8 porque o saldo de gols mostra a eficiência do time em marcar gols, o que implica em maior chance de vitória;

- x_9, x_{10}, x_{11}, x_{12} relacionadas aos cartões amarelos e vermelhos foram colocadas, pois se verifica a partir de dados das súmulas dos jogos, que o time com maior número de cartões tem grande chance de perder a próxima partida;

- e por último as variáveis x_{13} e x_{14}, visto que o time que teve mais escanteios, isto é, mais bolas alçadas na grande área, possui maior chance de gols.

3.3 MODELO

A hipótese do problema é a probabilidade de um jogo de futebol ter um dos três resultados (vitória do time mandante, empate ou vitória do time visitante), usando-se as quatorze variáveis explicitadas na seção anterior. Ou ainda, a hipótese é a probabilidade $m_\theta^{(i)}(x)$ do jogo de futebol pertencer a um dos resultados $i \in \{1, 2, 3\}$ é dada pelo modelo logístico

$$m_\theta^{(i)}(x) = \frac{1}{1 + e^{-\theta^T x}},$$

onde $\theta = (\theta_0, \theta_1, \ldots, \theta_{14}) \in \mathbb{R}^{15}$ é o vetor de parâmetros do modelo e $x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$.

Na primeira parte dividiu-se os dados dos jogos do campeonato de futebol até a j-ésima rodada, em 80% para treinamento e 20% para teste. Após, separou-se os dados na matriz X (que contém as variáveis dependentes) e y (com a variável resposta). Isso tanto para os dados de treinamento quanto para os de teste.

Em seguida modificou-se a variável resposta em três variáveis y_1, y_2, e y_3, pois a variável não é binária. Daí treinou-se um modelo $(m_\theta^{(i)}(x), i \in \{1, 2, 3\})$ para cada classe (vitória em casa, empate e vitória fora de casa). Após, fez-se a calibração dos parâmetros (θ), com a função fminunc do Octave, minimizando a função custo e calculamos o gradiente. Depois, estimou-se a probabilidade de acerto do modelo para cada classe i.

Por último calculou-se a taxa de acerto do modelo, usando o método One-vs-All, ou seja, verificamos a probabilidade em classificar corretamente o resultado de futuros jogos, conforme Subseção 2.3.1 do Capítulo 2.

3.4 RESULTADOS

Os dados são separados da seguinte maneira. Os 80% primeiros jogos do campeonato são destinados a dados de treinamento e os outros 20% dos jogos restantes são para dados de teste. Por exemplo, na 38º rodada do campeonato brasileiro de futebol de 2016, os 74 primeiros jogos são destinados aos dados de treinamento e os 296 últimos jogos para os dados de teste.

Logo após, faz-se a calibração dos parâmetros do modelo a partir dos dados de treinamento (até a 37º rodada). Para isso, usa-se a função fminunc do Octave para minimização de $f(\theta)$.

- $\theta_1 = (0.052792, -0.055634, 0.037041, -0.213204, 0.273918, 0.016475, 0.379946, -0.008066, -0.010336, -0.026599, 0.021269, -0.101654, 0.046058, 0.007559, -0.002586)$
- $\theta_2 = (-0.811197, 0.063573, -0.067846, 0.189929, -0.116299, 0.011633, -0.351742, 0.054331, -0.062862, -0.003092, -0.008081, 0.118317, -0.009630, -0.003104, 0.001661)$
- $\theta_3 = (-1.517057, 0.016315, 0.015415, 0.098236, -0.240250, -0.059633, -0.148089, -0.041307, 0.071331, 0.044573, -0.025140, 0.028795, -0.041806, -0.009674, 0.001598)$

Com os parâmetros calibrados e o modelo treinado, fez-se as previsões dos jogos da 38º rodada. Na tabela 1 veja a taxa de acerto do modelo. A taxa de acerto do modelo é calculado pela equação 3.1.

$$\text{Taxa de Acerto} = \frac{\#\text{ amostras classificadas corretamente}}{\#\text{ total de amostras}}. \tag{3.1}$$

Tabela 1: Taxa de acerto do modelo.

	MODELO
$m_\theta^{(1)}(x)$ - VITÓRIA DO TIME MANDANTE	55%
$m_\theta^{(2)}(x)$ - EMPATE	66%
$m_\theta^{(3)}(x)$ - VITÓRIA DO TIME VISITANTE	77%
ONE VS ALL	55%

Neste passo se faz as previsões para a 38º rodada do campeonato brasileiro de 2016, ou seja, as previsões da variável resposta ($\hat{y}$). Também se calcula as probabilidades de cada modelo $m_\theta^{(i)}$, $i \in \{1, 2, 3\}$. Veja as previsões e probabilidades na tabela 2.

Tabela 2: Probabilidade de cada modelo $m_\theta^{(i)}$ e previsão por partida da 38º rodada do campeonato brasileiro de 2016.

PARTIDA	$m_\theta^{(1)}$	$m_\theta^{(2)}$	$m_\theta^{(3)}$	Previsão
GRE x BOT	41.0%	15.2%	**41.3%**	3
CAP x FLA	**49.6%**	9.0%	44.5%	1
COR x CRU	39.7%	16.3%	**42.5%**	3
VIT x PAL	20.8%	8.8%	**82.1%**	3
CHA x CAM	**51.9%**	8.0%	44.0%	1
FLU x INT	**77.6%**	10.0%	10.5%	1
PON x CFC	**72.3%**	5.3%	24.8%	1
SPT x FIG	**70.8%**	26.9%	6.2%	1
SAN x AME	**78.9%**	61.8%	1.0%	1
SPA x STA	**79.2%**	22.6%	4.2%	1

Neste passo, calcula-se a taxa de acerto do modelo proposto, usando-se o método One-vs-All, conforme Subseção 2.4.1 do Capítulo 2. No gráfico 3 mostra-se a evolução do modelo com relação essa taxa de acerto do modelo.

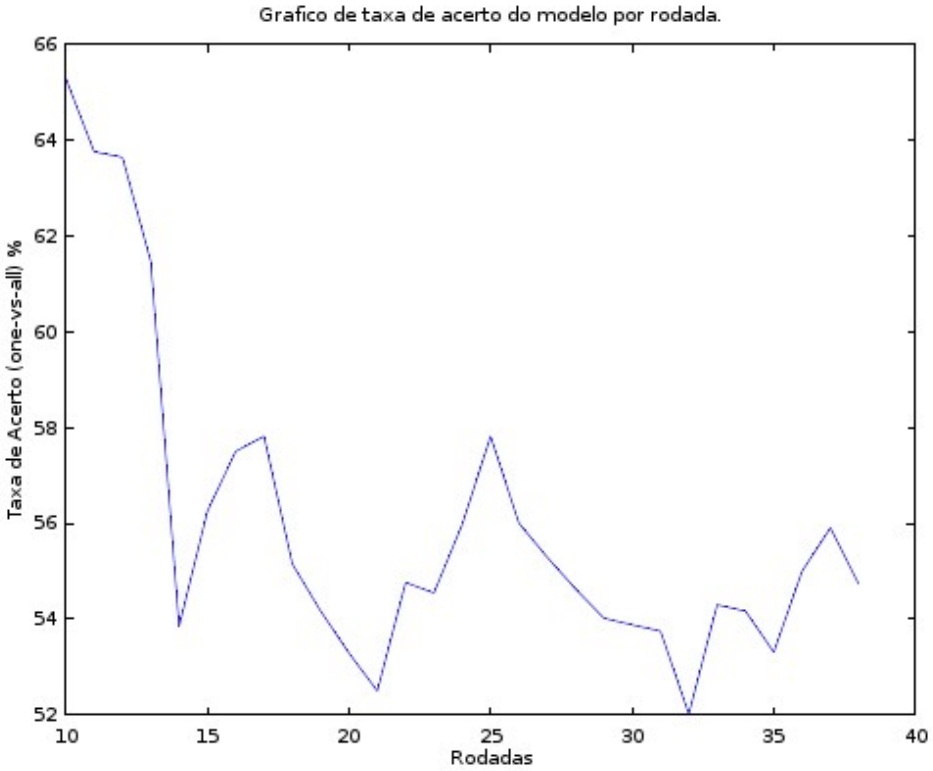

Figura 3: Taxa de acerto do modelo (One-vs-All) por rodada.

No final, realiza-se uma análise de comparação com resultados oficiais da CBF e com modelos usados em alguns sites[1] de previsões da internet. Comparando os resultados oficiais da CBF, verifica-se que o modelo proposto teve acerto em 6 em 10 jogos da 38º rodada. Os sites comparados tiveram acertos em: Chance de Gols (6 de 10 jogos), Rivalo (1 de 10 jogos), Sportingbet (1 de 10 jogos), Bet365 (1 de 10 jogos) e Betmotion (1 de 10 jogos). Portanto, o modelo desse trabalho teve uma taxa de acerto aceitável comparado com estes sites da internet.

[1]Link dos sites comparados: `http://chancedegol.uol.com.br/`, `https://www.rivalo.com`, `https://br.sportingbet.com/`, `http://www.bet365.com/` e `https://www.betmotion.com/br/`.

4 CONCLUSÕES

O modelo proposto obteve bons resultados, com relação a taxa de acerto e em comparação com outros modelos, usados em sites da internet. A taxa de acerto, por rodada, usando One-vs-All, ficou estável entre 48% e 57% na maioria das rodadas. Em comparação aos sites, o modelo obteve a quantidade de acerto equivalente ao melhor modelo entre os sites, para a 38º rodada do campeonato brasileiro de 2016.

Para trabalhos futuros, pretendemos fazer novos modelos com outros métodos no intuito de melhorar a taxa de acerto do modelo. Buscamos estimar as probabilidades do time ser campeão, se classificar para os campeonatos: Taça Libertadores da América e Copa Sul-americana, e de ser rebaixado. Também pretendemos estender este estudo para outros campeonatos de futebol.

Referências

ALPHA21. **Como fazer uma Regressão Logística**. Outubro 2016. [Online; accessed 04-outubro-2016]. Disponível em: <http://analise-estatistica.pt/2015/12/como-fazer-regressao-logistica.html>.

BITTENCOURT, H. R. Regressão logística politômica: revisão teórica e aplicações. **Acta Scientiae**, v. 5, n. 1, p. 77–86, 2012.

EATON DAVID BATEMAN, S. H. J. W.; WEHBRING, R. **GNU Octave version 4.0.0 manual: a high-level interactive language for numerical computations**. [s.n.], 2015. Disponível em: <http://www.gnu.org/software/octave/doc/interpreter>.

Equipe Estatcamp. **Software Action**. São Carlos - SP, Brasil, 2014. Disponível em: <http://www.portalaction.com.br/>.

FARIAS, F. **Análise e Previsao de Resultados de Partidas de Futebol. 2008. 78 p**. Tese (Doutorado) — Dissertaçao (Mestrado em Estatıstica), Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2008.

FIGUEIRA, C. V. Modelos de regressão logística. 2006.

MCCULLOUGH, P.; NELDER, J. A. **Generalized linear models**. [S.l.]: London: Chapman & Hall,, 1989.

PAULA, G. A. **Modelos de regressão: com apoio computacional**. [S.l.]: IME-USP São Paulo, 2004.

SENN, M. **Using LaTeX for Your Thesis**. [S.l.], 2016 (accessed Agosto 9, 2016). Disponível em: <https://pt.wikipedia.org/wiki/Fun%C3%A7%C3%A3o_log%C3%ADstica>.

SMOLA, A.; VISHWANATHAN, S. Introduction to machine learning. **Methods in Molecular Biology**, v. 1107, p. 105–128, 2014. ISSN 10643745.

(UFMA), P. de P.-G. **Regressão Logística**. Outubro 2016. [Online; accessed 04-outubro-2016]. Disponível em: <http://www.pgsc.ufma.br/arquivos/apostilaregressaologistica-.pdf>.

ANEXO A – Códigos para fazer a previsão dos resultados dos jogos da j-ésima rodada de campeonato que ocorre com sistema de pontos corridos.

```
%%=============================================================
%% Regressão logística multinomial - Sub-Rotina 1
%%=============================================================

%%-------------------------------------------------------------
%% Entrada dos dados e divisão em dados de teste e treinamento
%%-------------------------------------------------------------

%% Inicialização \\
clear ; close all; clc \\

%% Mudando o diretório para onde estão os dados
pwd
cd C:\Users\Fillipe Rafael\Desktop\modelo IC\Futebol
pkg load all % para carregar o pacote io que importa dados do excel.

%% Carregando dados
% Conjunto de dados "futbras": Consiste de 370 amostras, ou seja, de
% 37 rodadas de dados do campeonato brasileiro, onde se acrescenta
% mais dados após cada rodada.
% Vitória do Time A = 1; Empate = 2; Vitória do Time B = 3

fprintf(Carregando os dados e organizando.);
```

```
dados = xlsread(futbras); % Todos os dados

%%-----------------------------------------------------------------
%% Dividindo as quantidades de dados para treinamento e teste
%% 80\% dos dados para treinamento e 20\% para teste

nd = (size(dados)(1)-10)*0.2; % número de jogos para teste
md = (size(dados)(1)-10)-nd; % número de jogos para teste

%%-----------------------------------------------------------------
%% Dados de teste.
dados1 = [zeros(nd,size(dados)(2))];
dados1(1:nd,1:size(dados)(2)) =
dados(size(dados)(1)-9-nd:size(dados)(1)-10,1:size(dados)(2));

% Xt - X de teste e yt - y de teste

% Xt = dados1(:, [1,2,3,4,5,6,7,8]);
% yt = dados1(:, size(dados)(2)); % MODELO 1

% Xt = dados1(:, [1,2,3,4,5,6,7,8,9,10,11,12,13,14])
% yt = dados1(:, size(dados)(2)); % MODELO 2

Xt = dados1(:, [1,2,7,8,15,16,17,18]);
yt = dados1(:, size(dados)(2)); % MODELO 3

%%-----------------------------------------------------------------
%% Dados de treinamento.
dados2 = [zeros(md,size(dados)(2))];
dados2(1:md,1:size(dados)(2)) = dados(1:md,1:size(dados)(2));

% X - X de treinamento e y - y de treinamento

% X = dados2(:, [1,2,13,14,15,16,21,22]);
% y = dados2(:, size(dados)(2)); % modelo do professor - MODELO 1
```

```
% X = dados2(:, [1,2,13,14,15,16,21,22,23,24,25,26,27,28]);
% y = dados2(:, size(dados)(2)); % MODELO 2

X = dados2(:, [1,2,21,22,29,30,31,32]);
y = dados2(:, size(dados)(2)); % MODELO 3

%%=================================================================
%% Regressão logística multinomial - Sub-Rotina 2
%%=================================================================

%%-----------------------------------------------------------------
%% Com os dados de treinamento calcular o theta respectivo
%%-----------------------------------------------------------------

% Arruma-se dados para cada um dos modelos com dados testes, ou seja,
% arruma-se y para cada categoria y_i, onde i é o índice do resultado
% do jogo (Home Win, Draw e Home Away).

%%-----------------------------------------------------------------
% Arrumar dados de tratamento para dados binários conforme espécie.
% y1: dados com 1s para resultado de jogo Home Win e 0s para os demais.
% y2: dados com 1s para resultado de jogo Draw e 0s para os demais.
% y3: dados com 1s para resultado de jogo Home Away e 0s para os demais.

my = size(y)(1);
y1 = ones(my, 1); y2 = ones(my, 1); y3 = ones(my, 1);

for c = 1:my
     if y(c) ~= 1;
        y1(c) = 0;
     else
        y1(c) = 1;
     endif
```

```
     if y(c) ~= 2;
        y2(c) = 0;
     else
        y2(c) = 1;
     endif

      if y(c) ~= 3;
        y3(c) = 0;
     else
        y3(c) = 1;
     endif
endfor

%%-------------------------------------------------------------

% Configuração da matriz de dados de forma apropriada, e adicionar
% o termo intercepto
[m, n] = size(X);

% Adicionando o termo intercepto para X e X_test
X = [ones(m, 1) X];

% Parâmetros inciais
initial_theta = zeros(n + 1, 1);

% Carregando a função costFunction
%% --------------- Função Custo (cost) e Gradiente --------------
function [J, grad] = costFunction(theta, X, y)
% COSTFUNCTION computa a função custo e gradiente da regressão logística
% J = COSTFUNCTION(theta, X, y) calcula o custo usando theta como os
% parâmetros da regressão logística e o gradiente do cost

m = length(y); % quantidade de dados

J = 0;
```

```
grad = zeros(size(theta));

h = sigmoid(X*theta);
% J = (1/m)*sum(-y .* log(h) - (1 - y) .* log(1-h));
J = (1/m)*(-y* log(h) - (1 - y)* log(1-h));
grad = (1/m)*X*(h - y);
end
%%-------------------------------------------------------------

% Compute and display initial cost and gradient
[cost1, grad1] = costFunction(initial_theta, X, y1);
[cost2, grad2] = costFunction(initial_theta, X, y2);
[cost3, grad3] = costFunction(initial_theta, X, y3);

fprintf(Cost do theta inicial (zeros): f\n, cost1);

fprintf(Gradiente do theta inicial (zeros) com y1: \n);
fprintf( f \n, grad1);

fprintf(Gradiente do theta inicial (zeros) com y2: \n);
fprintf( f \n, grad2);

fprintf(Gradiente do theta inicial (zeros) com y3: \n);
fprintf( f \n, grad3);

% fprintf(\nPrograma pausado. Pressione enter para continuar.\n);
% pause;

%%-------------------------------------------------------------
%% Otimização usando a função fminunc
%%-------------------------------------------------------------

%% --------------- Otimização usando fminunc --------------------
% Usando a função interna (fminunc) para encontrar a parâmetros
% theta ideal de cada espécie i, i=1,2,3.
```

```
%  Definir opções para a função fminunc
options = optimset(GradObj, on, MaxIter, 400);

%  Rodando fminunc para obter o theta ideal
%  Essa função retorna theta e cost
[theta1, cost1] = fminunc(@(t)(costFunction(t, X, y1)),
        initial_theta, options);
[theta2, cost2] = fminunc(@(t)(costFunction(t, X, y2)),
        initial_theta, options);
[theta3, cost3] = fminunc(@(t)(costFunction(t, X, y3)),
        initial_theta, options);

Imprimir theta para a tela
fprintf(Cost de theta encontrado pela fminunc modelo Home: f\n, cost1);
fprintf(theta 1: \n);
fprintf( f \n, theta1);

fprintf(Cost de theta encontrado pela fminunc modelo Draw: f\n, cost2);
fprintf(theta 2: \n);
fprintf( f \n, theta2);

fprintf(Cost de theta encontrado pela fminunc modelo Awin: f\n, cost3);
fprintf(theta 3: \n);
fprintf( f \n, theta3);

fprintf(\nPrograma pausado. Pressione enter para continuar.\n);
pause;

%%-------------------------------------------------------------
%% Otimização usando a função fminunc - MODELO COM REGULARIZAÇÃO
%%-------------------------------------------------------------

%% --------------- Otimização usando fminunc --------------------
%  Usando a função interna (fminunc) para encontrar a parâmetros
```

```
% theta ideal de cada espécie i, i=1,2,3.

options = optimset(GradObj, on, MaxIter, 400);

%  Rodando fminunc para obter o theta ideal
%  Essa função retorna theta e cost
[theta1, cost1] = fminunc(@(t)(lrCostFunction(t, X, y1, lambda=1.0)),
        initial_theta, options);
[theta2, cost2] = fminunc(@(t)(lrCostFunction(t, X, y2, lambda=1.0)),
        initial_theta, options);
[theta3, cost3] = fminunc(@(t)(lrCostFunction(t, X, y3, lambda=1.0)),
        initial_theta, options);

% Imprimir theta para a tela
fprintf(Cost de theta encontrado pela fminunc modelo da espécie 1:
        f\n, cost1);
fprintf(theta 1: \n); fprintf( f \n, theta1);

fprintf(Cost de theta encontrado pela fminunc modelo da espécie 2:
        f\n, cost2);
fprintf(theta 2: \n); fprintf( f \n, theta2);

fprintf(Cost de theta encontrado pela fminunc modelo da espécie 3:
        f\n, cost3);
fprintf(theta 3: \n); fprintf( f \n, theta3);

fprintf(\nPrograma pausado. Pressione enter para continuar.\n);
pause;

%%=============================================================
%% Regressão logística multinomial - Sub-Rotina 3
%%=============================================================

%%-------------------------------------------------------------
%% Teste da precisão do modelo de regressão logística para os dados teste
```

```
%%-------------------------------------------------------------

%% ------------------ Valor Predito e Precisão ------------------
% Após estimados os parâmetros você gostaria de usá-lo para prever
% resultados de dados futuros ou não visíveis. Nesta etapa você usará
% o modelo de regressão logística para classificar cada jogo do campeonato
% no seu resultado (Home Win, Draw, Home Away).

%  Além disso, você irá calular a precisão de treinamento do modelo
% e com o conjunto de dados teste (dado1).

%%-------------------------------------------------------------
% Arrumar dados de teste para dados binários conforme espécie.
% Arrumar dados de tratamento para dados binários conforme espécie.
% yt1: dados com 1s para resultado de jogo Home Win e 0s para os demais.
% yt2: dados com 1s para resultado de jogo Draw e 0s para os demais.
% yt3: dados com 1s para resultado de jogo Home Away e 0s para os demais.

myt = size(yt)(1);
yt1 = ones(myt, 1); yt2 = ones(myt, 1); yt3 = ones(myt, 1);

for c = 1:myt
     if yt(c) ~= 1;
        yt1(c) = 0;
     else
        yt1(c) = 1;
     endif

     if yt(c) ~= 2;
        yt2(c) = 0;
     else
        yt2(c) = 1;
     endif

      if yt(c) ~= 3;
```

```
        yt3(c) = 0;
     else
        yt3(c) = 1;
     endif
endfor

%%-------------------------------------------------------------

%% ---------------- Calculo da Precisão dos Dados --------------

% Carregando a função sigmoid
%% ----------------- Função Logística - Sigmóide ----------------
function g = sigmoid(z)
#SIGMOID Calcula função logística
%   J = SIGMOID(z) calula a sigmoid de z.

% Instrução: Calcule a sigmóide de cada valor de z (z deve ser uma
% matriz, vetor ou escalar).

g = 1 ./ (1 + e.^-z);
end
%%-------------------------------------------------------------

% Calcula a precisão do dados de teste (dados1)
Xt = [ones(nd, 1) Xt];

p1 = sigmoid(Xt*theta1)>=0.5;
p2 = sigmoid(Xt*theta2)>=0.5;
p3 = sigmoid(Xt*theta3)>=0.5;

% p_i: cada um dos modelos aplicados aos dados de teste.
% p_i == yt, compara compara dados aplicados aos dados de teste
% com dados originais.
fprintf(Taxa de Acerto Modelo 1: f\n, mean((p1 == yt1)) * 100);
fprintf(Taxa de Acerto Modelo 2: f\n, mean((p2 == yt2)) * 100);
```

```
fprintf(Taxa de Acerto Modelo 3: f\n, mean((p3 == yt3)) * 100);

fprintf(\nPrograma pausado. Pressione enter para continuar.\n);
pause;

%%=============================================================
%% Regressão logística multinomial - Sub-Rotina 3
%%=============================================================

%%-------------------------------------------------------------
%% Teste da precisão do modelo de regressão logística para os dados teste
%%-------------------------------------------------------------

%% ------------------- Valor Predito e Precisão -----------------
% Após estimados os parâmetros você gostaria de usá-lo para prever
% resultados de dados futuros ou não visíveis. Nesta etapa você usará
% o modelo de regressão logística para classificar cada jogo do
% campeonato no seu resultado (Home Win, Draw, Home Away).

%  Além disso, você irá calular a precisão de treinamento do modelo e
% com o conjunto de dados teste (dado1).

%%-------------------------------------------------------------
% Arrumar dados de teste para dados binários conforme espécie.
% Arrumar dados de tratamento para dados binários conforme espécie.
% yt1: dados com 1s para resultado de jogo Home Win e 0s para os demais.
% yt2: dados com 1s para resultado de jogo Draw e 0s para os demais.
% yt3: dados com 1s para resultado de jogo Home Away e 0s para os demais.

myt = size(yt)(1);
yt1 = ones(myt, 1); yt2 = ones(myt, 1); yt3 = ones(myt, 1);

for c = 1:myt
     if yt(c) ~= 1;
        yt1(c) = 0;
```

```
      else
         yt1(c) = 1;
      endif

      if yt(c) ~= 2;
         yt2(c) = 0;
      else
         yt2(c) = 1;
      endif

       if yt(c) ~= 3;
         yt3(c) = 0;
      else
         yt3(c) = 1;
      endif
endfor

%%---------------------------------------------------------------

%% ---------------- Calculo da Precisão dos Dados ---------------

% Carregando a função sigmoid
%% ----------------- Função Logística - Sigmóide ----------------
function g = sigmoid(z)
#SIGMOID Calcula função logóstica
%   J = SIGMOID(z) calula a sigmoid de z.

% Instrução: Calcule a sigmóide de cada valor de z (z deve ser uma
% matriz, vetor ou escalar).

g = 1 ./ (1 + e.^-z);
end
%%---------------------------------------------------------------

% Calcula a precisão do dados de teste (dados1)
```

```
Xt = [ones(nd, 1) Xt];

p1 = sigmoid(Xt*theta1)>=0.5;
p2 = sigmoid(Xt*theta2)>=0.5;
p3 = sigmoid(Xt*theta3)>=0.5;

% p_i: cada um dos modelos aplicados aos dados de teste.
% p_i == yt, compara compara dados aplicados aos dados de teste com
% dados originais.
fprintf(Taxa de Acerto Modelo 1: f\n, mean((p1 == yt1)) * 100);
fprintf(Taxa de Acerto Modelo 2: f\n, mean((p2 == yt2)) * 100);
fprintf(Taxa de Acerto Modelo 3: f\n, mean((p3 == yt3)) * 100);

fprintf(\nPrograma pausado. Pressione enter para continuar.\n);
pause;

%%=================================================================
%% Regressão logística multinomial - Sub-Rotina 4
%%=================================================================

%%-----------------------------------------------------------------
%% Método One-vs-All
%%-----------------------------------------------------------------

m1 = sigmoid(X*theta1);
m2 = sigmoid(X*theta2);
m3 = sigmoid(X*theta3);

%%-----------------------------------------------------------------
% a função argmax no Matlab ou max no Octave classifica a linha j
% (j = 1, 2, ..., 300), no resultado i (i = 1, 2, 3), ou seja,
% prevemos o resultado do jogo do campeonato brasileiro.

ova = ones(size(y)(1), 1);
for d = 1:size(y)(1)
```

```
    [c,i] = max([m1(d), m2(d), m3(d)]);
    ova(d) = i;
endfor

fprintf(Taxa de Acerto Modelo One-vs-All: #f\n, mean((ova == y)) * 100);

%%=============================================================
%% Regressão logística multinomial - Sub-Rotina 5
%%=============================================================

%%-------------------------------------------------------------
%% Novos Dados - Próxima Rodada
%%-------------------------------------------------------------

%% Dados da j-ésima rodada.
dados3 = [zeros(10,size(dados)(2))];
dados3(1:10,1:size(dados)(2)) =
        dados(size(dados)(1)-9:size(dados)(1),1:size(dados)(2));
% Xn = dados3(:, [1,2,3,4,5,6,7,8]); % MODELO 1
% Xn = dados3(:, [1,2,3,4,5,6,7,8,9,10,11,12,13,14]); % MODELO 2
Xn = dados3(:, [1,2,7,8,15,16,17,18]); % MODELO 3

Xn = [ones(size(Xn)(1), 1) Xn];

m1n = sigmoid(Xn*theta1);
m2n = sigmoid(Xn*theta2);
m3n = sigmoid(Xn*theta3);

fprintf(Probabilidade de acerto para cada jogo (coluna 1 - Home Win,
        coluna 2 - Draw e coluna 3 - Away Win):);
[fix(m1n*100), fix(m2n*100), fix(m3n*100)]
fprintf(\nPrograma pausado. Pressione enter para continuar.\n);
pause;

%%-------------------------------------------------------------
```

```
% a função argmax no Matlab ou max no Octave classifica a linha j
% da próxma rodada do campeonato brasileiro, no resultado (i = 1, 2, 3),
% ou seja, prevemos o resultado dos jogos da próxima rodada do
% campeonato brasileiro.

new = zeros(10, 1); % ova = ones(size(y)(1), 1);
por = zeros(10, 1); % ova = ones(size(y)(1), 1);
for d = 1:10 % for d = 1:size(y)(1)
     [c,i] = max([m1n(d), m2n(d), m3n(d)]);
     new(d) = i;
     por(d) = c;
endfor

fprintf(Previsões para próxima rodada (coluna 1) e probabilidade de
          acerto (coluna 2): );
[fix(new),fix(por*100)]
%%-----------------------------------------------------------------
```