Regressao Linear e Logística

Joao Paulo Pordeus - LOGIA - UFC

March 23, 2016

1 Variáveis

1.1 Univariada

O objetivo da regressão linear univariada é correlacionar dois conjuntos de variáveis $\{x_i\}$ e $\{y_i\}$ através de uma equação linear.

Uma equação tem a forma

$$\overline{y_i} = w_0 + w_1 x_i \tag{1}$$

Onde x_i é a variável explanatória, e $\overline{y_i}$ é o valor obtido pelo modelo, que desejamos que seja o mais próximo possível de y_i. Os parâmetros obtidos w_0 e w_1 determinam portanto uma reta.

1.2 Multivariada

Na regressão multivariada, levamos em consideração diversas variáveis explicativas x_{ij} influenciando $\overline{y_i}$ ao mesmo tempo. Na regressão multivariada, a estimação da i-ésima amostra é dada por:

$$\overline{y_i} = w_0 + x_{i1} w_1 + x_{i2} w_2 + ... + x_{im} w_m \tag{2}$$

em que x_{ab} e o b-esimo atributo da a-esima amostra. Seja a amostra x_i representada por $[1, x_{i1}, x_{i2}, ..., x_{im}]^T$, onde o 1 e adicionado ao vetor por conveniência. Construímos uma matriz X de atributos onde $X = [x_1, x_2, ..., x_n]^T$. Além disso, seja $w = [w_1, w_2, ..., w_n]^T$ o vetor de pesos. Dessa forma, podemos escrever a saída do modelo por

$$\bar{y_i} = w^T x_i \tag{3}$$

2 Métodos

Todos os métodos utilizam em sua concepção o erro quadrático médio J, definido a partir de

$$J(w) = \frac{1}{2n} \sum_{i=1}^{n} e_i^2 \tag{4}$$

Onde $e_i = y_i - \overline{y_i} = y_i - w^T x_i$. Inserimos o 2 no quociente por conveniência, pois ele simplifica a fórmula obtida mais adiante quando derivarmos as expressões.

2.1 Gradiente descendente

O método do gradiente descendente baseia-se em alterar o vetor w levemente a cada iteracao. Para tanto, decrementaremos w de uma pequena fração do gradiente da função J no ponto w.

$$w = w - \alpha\vec{\nabla}J(w) \tag{5}$$

Onde α é um valor convenientemente pequeno (mais sobre esse valor na seção 4.1). Lembrando que:

$$\vec{\nabla}J(w) = (\frac{\partial J}{\partial w_0}, \frac{\partial J}{\partial w_1}, ..., \frac{\partial J}{\partial w_m}) \tag{6}$$

Em que $\partial J/\partial x$ e a derivada parcial de J no ponto x. Derivando J, encontramos:

$$w_i = w_i - \alpha\frac{1}{n}\sum_{i=1}^{n} e_i x_i \tag{7}$$

A cada iteração desse processo, o vetor w normalmente fornece uma aproximação melhor para a regressão. Assim, repetimos esse processo um certo número de vezes. Chamamos essa quantidade de número de *épocas*. Fazemos isso até que a solução seja boa o suficiente, ou seja, o erro quadrático médio esteja próximo de zero, ou que a mudança no valor de w deixe de melhorar a solução. (Mais detalhes na seção 4.2)

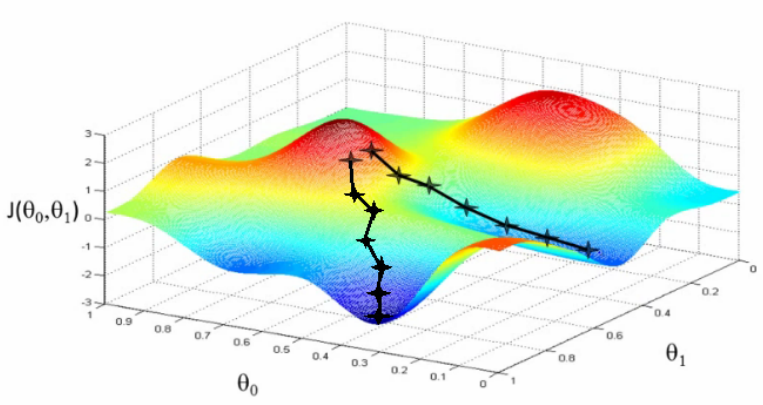

Figure 1: Cada ponto no traçado é um valor obtido atraves da mudança de w. Regiões vermelhas representam picos e azuis, vales. Observe que o algoritmo está saindo de uma região de pico para chegar numa região de valor mínimo.

2.2 Gradiente descendente estocástico

O gradiente descendente estocástico traca uma abordagem um pouco diferente do método tradicional. Em vez de utilizar a media dos valores de $e_i x_i$, ele utiliza apenas um valor $e_i x_i$ por iteração. A cada época, uma permutação dos valores

de $e_i x_i$ é gerada, e essa sequência é utilizada para atualizar os valores de w. Assim, a formula de atualização dos pesos passa a ser:

$$w = w - \alpha e_i x_i \tag{8}$$

2.3 Em lote - Batch

Sendo $Y = [y_1, y_2, ..., y_n]^T$ e $\overline{Y} = [\overline{y_1}, \overline{y_2}, ..., \overline{y_n}]^T$, nosso problema é reduzir o valor do erro quadratico, ou seja, minimizar a função:

$$J(w) = \frac{1}{2}(Y - \overline{Y})^T (Y - \overline{Y}) \tag{9}$$

Lembrando que $\overline{Y} = Xw$ e fazendo $\partial J / \partial w = 0$, encontramos:

$$w = (X^T X)^{-1} X^T Y \tag{10}$$

3 Regressão Logística

A regressão logística usa a regressão linear para realizar classificação. Transformamos a saída da regressão linear utilizando a *função logística*:

$$f(x) = e^x/(e^x + 1) = 1/(1 - e^{-x}) \tag{11}$$

Cujo comportamento é trazer rapidamente a 1 numeros positivos e a 0 os numeros negativos, ou seja, a valores binários, que podem ser usados como operadores *lógicos*.

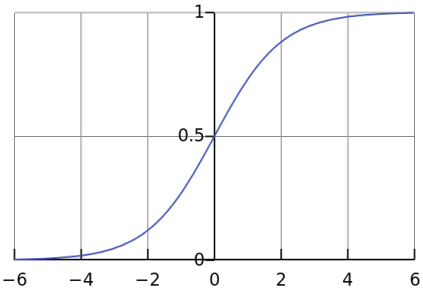

Figure 2: Gráfico da função logística

As estimações para a regressão passam a ser:

$$\overline{y_i} = f(w^T x_i) \tag{12}$$

Infelizmente a função de erro J que usamos até aqui nao funciona mais, porque a inserção da função logística torna o gradiente não convexo. Trocaremos para uma nova função de custo:

$$J(w) = \frac{1}{2n} \sum_{i=1}^{n} C(i) \tag{13}$$

$$C(i) = -y_i ln(\overline{y_i}) - (1 - y_i) ln(1 - \overline{y_i}) \tag{14}$$

Essa nova função tem o seguinte comportamento: quando y_i é 1, $C(i)$, será $-ln(\overline{y_i})$. Se y_i for 0, então $C(i)$ será $ln(1 - \overline{y_i})$. Em ambos os casos, $C(i)$ se aproxima de zero quando $\overline{y_i}$ se aproxima de y_i, logo $C(i)$ nos fornece uma noção coerente de erro, fazendo sentido querer diminuir seu somatório.

Finalmente, a nova derivada parcial passa a ser

$$\frac{\partial J}{\partial w_i} = -\frac{1}{n} \sum_{i=1}^{n} (\overline{y_i} - y_i) x_i \tag{15}$$

Vale ressaltar tambem que apesar do funcionamento da regressão logística ser binário com relação a identificação da classe, podemos utilizá-la para mais de duas. Para identificar uma certa classe K_1 usando amostras de outras duas(ou mais)classes K_2 e K_3, basta identificá-las todas como do tipo 0, enquanto apenas a classe K_1 recebe a saída 1. A seguir, para identificar elementos de K_2, fazemos os elementos de K_2 terem saída 1 e de K_1 e K_3 saída 0.

4 Hiperparâmetros

Dois hiperparâmetros merecem atenção na regressão, o α e o número de épocas, cada um influenciando de certa maneira o algoritmo. Não existem valores ideais para esses hiperparâmetros, apesar de existir abordagens que tentam melhorar seus usos. Aqui, a melhor saída é a experimentação. Valores típicos de números de época giram em torno de 1000, enquanto que α fica em torno de 0.01

4.1 Alfa

O valor de α é o valor multiplicado ao valor que pretende ser incrementado ao vetor w. Ele determina o tamanho do salto que é dado na função que representa o erro (veja figura 1). Se α for grande, os saltos serão grandes e o algoritmo rapidamente chega ao valor mínimo. Todavia, ao se aproximar da solucao ótima, o algoritmo não consegue chegar ao ponto exato, pois ele não é capaz de dar saltos pequenos o suficiente para chegar a um ponto preciso.

Para entender melhor o problema, suponha que você deseja chegar a um ponto que está a 5 metros de distância, mas você só é capaz de dar passos de 7 metros. Se, assim como o algoritmo, você está sempre caminhando em direção a solução ótima, então você nunca chegará ao ponto desejado. De fato, ao dar seu passo, você estará a 2 metros do ponto. Mas ao se mover em direção a ele novamente, você voltará a posição inicial do problema.

Naturalmente, se α for pequeno, esse problema é resolvido, mas isso pode tornar o algoritmo lento, ou até virtualmente parar, se o incremento tornar-se zero.

4.2 Número de épocas

O número de épocas é o número de vezes que se repete o algoritmo. Em geral, a cada iteração, fica-se mais próximo da solução ótima. O número necessário de épocas é desconhecido, assim, dois problemas podem surgir: se o número de épocas utilizado for menor que o necessário, talvez não se chegue próximo o

suficiente da solução ótima; se por outro lado o número de épocas for maior que o necessário, o algoritmo irá perder tempo com iterações que nao fazem nada útil. Isso pode ser resolvido com o término do algoritmo caso a iteração nao melhore a solução, ou se a solução obtida já for considerada boa o suficiente.

5 Modelos não lineares

Os modelos lineares que vimos até aqui possuem fundamental importância teórica, porém pouca ou nenhuma aplicação prática. Os dados no mundo real dificilmente são *linearmente separáveis*, ou seja, não podem ser classificados simplesmente através da regressão logística. É todavia possível utilizar a regressão com modelos não lineares. Podemos por exemplo adicionar à equação 2 a combinação quadrática de $\{x_i\}$:

$$\overline{y_i} = w_0 + x_{i1}w_1 + ... + x_{im}w_m + x_{i1}^2 w_1' + x_{i2}^2 w_2' + ... + x_{in}^2 w_m' \qquad (16)$$

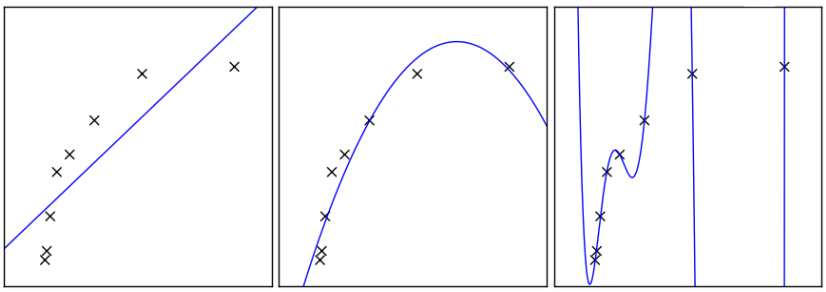

Figure 3: À esquerda, um modelo linear. No centro, um modelo quadrático. À direita, um modelo polinomial de grau 6. Note como o modelo acerta cada vez melhor os dados fornecidos

De fato, e possivel complicar arbitrariamente o modelo, aumentando indefinidamente sua expressividade. Essa complicacao, porem, nem sempre e desejada. A maioria dos modelos estudados daqui em diante serao nao lineares.

5.1 Overfitting

O primeiro problema advindo do aumento de complexidade é um problema inerente a todos os algoritmos de aprendizagem de máquina: chama-se *overfitting*. O problema ocorre quando a saída do modelo se aproxima demais aos dados da amostra, mas não aos dados *fora dela*. Dizemos que o erro de aprendizado é quase zero, mas mesmo assim o modelo não generaliza bem.

Observe a figura 4. À esquerda, temos o modelo obtido, em linha vermelha, quando o conjunto de treinamento é composto pelos quatro pontos dados. Mas ao final, o modelo não representa bem todos os dados da distribuição. De fato, um modelo linear, menos complexo, representaria melhor.

De modo semelhante, na figura 3, observando a amostra, parece claro que o modelo quadrático responde melhor a generalização, e que o modelo polinomial de grau 6, apesar de ter um erro quadrático menor, não parece nada natural.

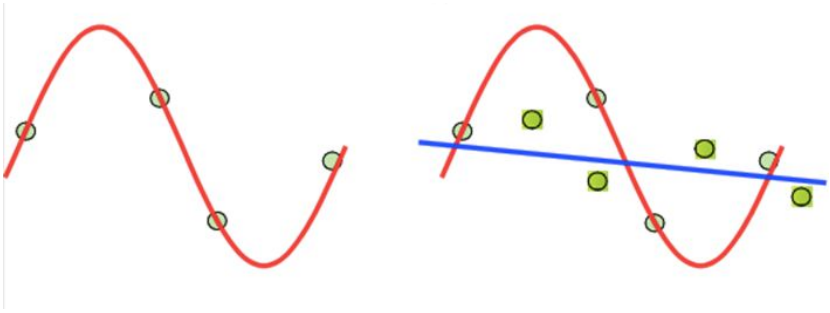

Figure 4: Nos dados da amostra, o modelo quadrático se aproxima muito bem (a esquerda). Porem, quando se incluem dados fora dela, modelo linear possui um erro menor (a direita)

Esse comportamento matemático em que o modelo se *esforça demais* para diminuir o erro, fazendo o gráfico dar muitas curvas para tocar os dados fornecidos é o que caracteriza o *overfitting*. Observe que o resultado do overfitting é um modelo que parece cada vez menos natural para explicar o fenômeno.

Assim, encontramos um problema dual na generalização: se o modelo é simples demais, o erro é grande demais. Se o modelo é complexo demais, as previsões erram muito.

Uma abordagem para resolver esse problema é dividir o conjunto de amostras em duas partes. A primeira parte, chamada de *conjunto de treinamento*, é utilizada para treinar o modelo, melhorando os coeficientes através das iterações. É importante notar que o erro quadrático médio medido nesse grupo sempre tenderá a um determinado valor, que é o ponto de máximo overfitting do modelo (que depende de sua complexidade), um valor que pode chegar a zero. Entretanto, isso não significa que o modelo generaliza bem. Para saber se o modelo generaliza bem, mediremos o erro quadratico médio do segundo grupo, chamado de *conjunto de teste*, que para o modelo, funciona como um dado desconhecido e novo. Se o erro do segundo grupo também cair, então isso implica que o modelo está aprendendo a prever a amostra. Eventualmente, o overfitting nos dados de treinamento fará o erro aumentar no conjunto de testes, o que representa o momento adequado para parar as iterações.

5.2 Regularização

Uma outra abordagem para diminuir o overfitting é a regularização. Quando temos muitos atributos, o modelo passa a ter muita liberdade para tentar produzir um overfitting. Se formos capazes de dizer quais atributos são menos importantes para o treinamento, então podemos tirar do modelo parte de sua liberdade.

Faremos isso alterando a equação 4 do erro:

$$J(w) = \frac{1}{2n}\sum_{i=1}^{n} e_i^2 + \lambda \sum_{i=1}^{m} w_i^2 \tag{17}$$

Nesse momento, essa equação não representa mais o erro, mas sim o *custo* que queremos minimizar. Esse custo e composto pelo erro, alem do tamanho

do vetor w, excetuando-se a componente w_0. A intencao e que tente-se reduzir os coeficientes ao maximo possivel sem que isso aumente o erro. Naturalmente, alguns coeficientes diminuirao mais que outros, esses se caracterizando como menos vitais para a generalizacao.

A derivada parcial de J com relacao a w_i e:

$$\frac{\partial J}{\partial w_j} = -\frac{1}{n}\sum_{i=1}^{n} e_i x_{ij} + \frac{\lambda}{n} w_j \tag{18}$$

Todavia, w_0 nao e afetado pela regularizacao, mantendo portanto sua atualizacao conforme a equacao 7