Log-linear Models

Jon Dehdari

December 7, 2015

Good Morning!

Some Preliminaries

Vector : In this context, a sequence of numbers. Eg.:

$$\mathbf{x} = \overrightarrow{x} = \langle 2.1, -8.5, 0.1 \rangle$$

$$\mathbf{y} = \overrightarrow{y} = \langle 1.1, 5.0, -4.4 \rangle$$

Some Preliminaries

Vector : In this context, a sequence of numbers. Eg.:

$$\mathbf{x} = \overrightarrow{x} = \langle 2.1, -8.5, 0.1 \rangle$$

$$\mathbf{y} = \overrightarrow{y} = \langle 1.1, 5.0, -4.4 \rangle$$

Dot Product : Multiplying corresponding elements of two vectors, then adding them all up (here, a.k.a. 'inner product')

$$\mathbf{x} \cdot \mathbf{y} = \mathbf{x}^{\mathbf{T}}\mathbf{y} = \sum_{i=1}^{n} x_i \times y_i = x_1\, y_1 + x_2\, y_2 + x_3\, y_3 \ldots$$

$$= (2.1 \times 1.1) + (-8.5 \times 5.0) + (0.1 \times -4.4)$$

$$= 2.31 + -42.5 + -0.44 = -\mathbf{40.63}$$

Some Preliminaries

Euler's Number : $e = 1 + \frac{1}{1} + \frac{1}{1 \cdot 2} + \frac{1}{1 \cdot 2 \cdot 3} + \cdots \approx \mathbf{2.71828}$

Some Preliminaries

Euler's Number : $e = 1 + \frac{1}{1} + \frac{1}{1 \cdot 2} + \frac{1}{1 \cdot 2 \cdot 3} + \cdots \approx \mathbf{2.71828}$

Matrix : A 2-dimensional vector

$$\boldsymbol{A} = \begin{bmatrix} 2.1 & -8.5 & 0.1 \\ 1.1 & 5.0 & -4.4 \end{bmatrix}$$

Some Preliminaries

Euler's Number : $e = 1 + \frac{1}{1} + \frac{1}{1 \cdot 2} + \frac{1}{1 \cdot 2 \cdot 3} + \cdots \approx \mathbf{2.71828}$

Matrix : A 2-dimensional vector

$$\boldsymbol{A} = \begin{bmatrix} 2.1 & -8.5 & 0.1 \\ 1.1 & 5.0 & -4.4 \end{bmatrix}$$

Tensor : In this context, an *n*-dimensional vector

Log-linear Models

$$P(y|\mathbf{x}) = \frac{e^{\boldsymbol{W}_y \cdot \mathbf{x}}}{Z}$$

$\leftarrow$ exponentiation helps ensure scores are positive

$\leftarrow$ **normalization constant**, to ensure the score of all possible outcomes sums to 1

$$= \frac{e^{\boldsymbol{W}_y \cdot \mathbf{x}}}{\sum_h e^{\boldsymbol{W}_h \cdot \mathbf{x}}}$$

$\leftarrow$ to get Z, we just add up scores from all possible outcomes

$$= \text{softmax}(\boldsymbol{W}_y \cdot \mathbf{x})$$

Log-linear Models

$$P(y|\mathbf{x}) = \frac{e^{\boldsymbol{W}_y \cdot \mathbf{x}}}{Z}$$

← exponentiation helps ensure scores are positive

← **normalization constant**, to ensure the score of all possible outcomes sums to 1

$$= \frac{e^{\boldsymbol{W}_y \cdot \mathbf{x}}}{\sum_h e^{\boldsymbol{W}_h \cdot \mathbf{x}}}$$

← to get Z, we just add up scores from all possible outcomes

$$= \mathrm{softmax}(\boldsymbol{W}_y \cdot \mathbf{x})$$

The input vector $\mathbf{x}$ includes an additional dummy value of 1.0 , called a **bias term**. This helps determine the offset of the linear separator.

Visualization

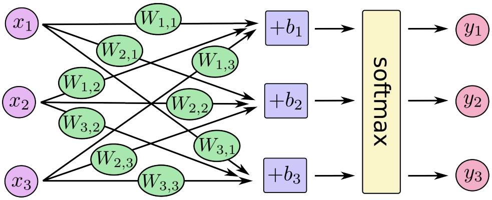
$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \text{softmax}\begin{pmatrix} W_{1,1}x_1 + W_{1,2}x_2 + W_{1,3}x_3 + b_1 \\ W_{2,1}x_1 + W_{2,2}x_2 + W_{2,3}x_3 + b_2 \\ W_{3,1}x_1 + W_{3,2}x_2 + W_{3,3}x_3 + b_3 \end{pmatrix}$$

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \text{softmax}\left(\begin{bmatrix} W_{1,1} & W_{1,2} & W_{1,3} \\ W_{2,1} & W_{2,2} & W_{2,3} \\ W_{3,1} & W_{3,2} & W_{3,3} \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} \right)$$

Visualization

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \text{softmax} \begin{pmatrix} W_{1,1}x_1 + W_{1,2}x_2 + W_{1,3}x_3 + b_1 \\ W_{2,1}x_1 + W_{2,2}x_2 + W_{2,3}x_3 + b_2 \\ W_{3,1}x_1 + W_{3,2}x_2 + W_{3,3}x_3 + b_3 \end{pmatrix}$$

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \text{softmax} \left(\begin{bmatrix} W_{1,1} & W_{1,2} & W_{1,3} \\ W_{2,1} & W_{2,2} & W_{2,3} \\ W_{3,1} & W_{3,2} & W_{3,3} \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} \right)$$



Courtesy of TensorFlow

Finding Good Values for **W**

- The weight matrix **W** constitutes the **parameters** of the model
- Then the main task is to find good values for the weight matrix **W**
- This is done by common **optimization** techniques, like L-BFGS for logistic regression

Terminology

- Logistic regression is over 50 years old

Terminology

- Logistic regression is over 50 years old
- It's sometimes also called a **maximum entropy** classifier (MaxEnt) and softmax regression, *inter alia*
- It also can be viewed as a **neural network** without any hidden layers

Terminology

- Logistic regression is over 50 years old
- It's sometimes also called a **maximum entropy** classifier (MaxEnt) and softmax regression, *inter alia*
- It also can be viewed as a **neural network** without any hidden layers
- The model is called a **log-linear model**. The following all have a log-linear model, but are trained differently:
  - **Logistic regression / MaxEnt / Softmax regression**
  - **Perceptron**
  - **Support vector machines** (SVMs)
  - **Conditional random fields** (CRFs)
  - **Linear discriminant analysis** (LDA)