Artificial Intelligence and Machine Learning Lecture #6 part 1

Amin Noroozi

University of Wolverhampton

a.noroozifakhabi@wlv.ac.uk

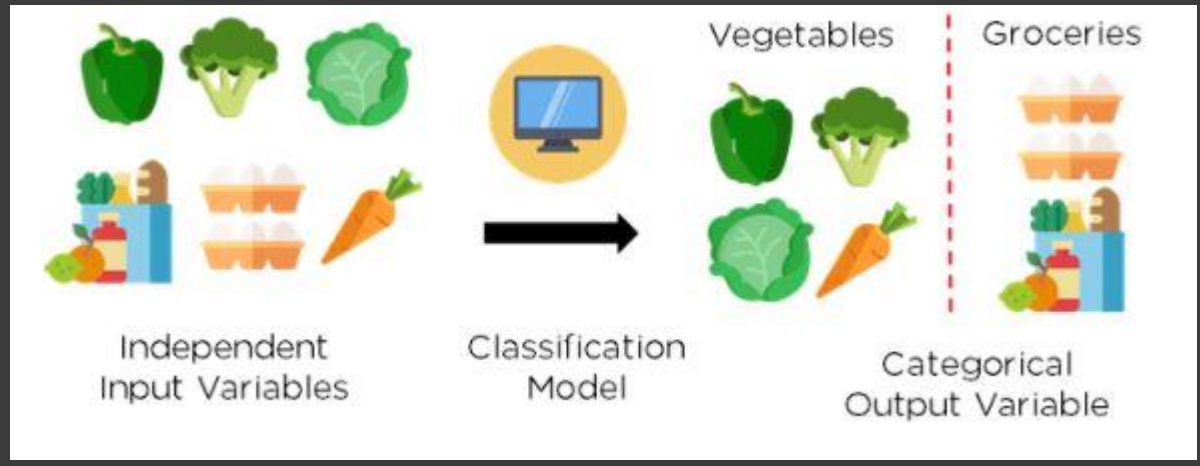
https://www.linkedin.com/in/amin-n-148350218/

❖ Classification

In regression, the target variable is continuous, but in classification the target variable is categorical

Regression => Prediction of house or stock price

Classification => Predicting whether a customer buys life insurance or not, which party a person is going to vote for: Democratic, republican, independent



❖ Logistic regression

In linear regression, we predict the *i*th output $y^{(i)}$ using the *i*th sample (row) in the dataset $x^{(i)}$ using the following formula:

$$y^{(i)} = \sum_j \theta_j x_j^{(i)} = \theta^T x^{(i)}$$

Or

$$y = h_\theta(x) = \theta^T x$$

❖ Logistic regression

However, the linear regression is not a good solution when the output is a binary-valued label, i.e. $y^{(i)} \in \{0,1\}$

To resolve this issue, in logistic regression, we do the following:

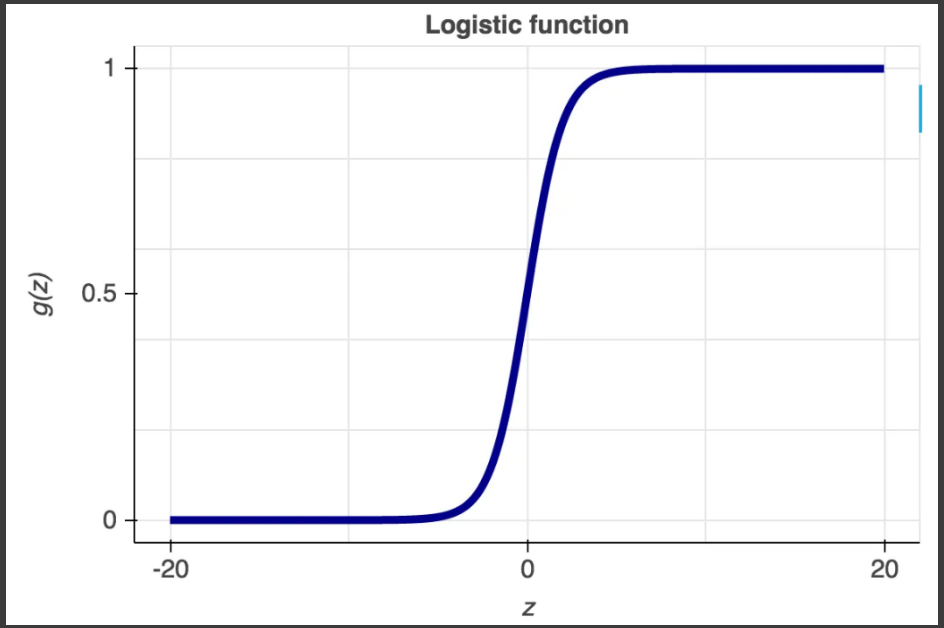
Step 1: we map the predicted output by linear regression into a probability value between 0 and 1 using the 'sigmoid' or 'logistic' function

$$h_\theta(\mathbf{x}) = g(\theta^\top \mathbf{x})$$

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

$$h_\theta(\mathbf{x}) = \frac{1}{1+e^{-\theta^\top \mathbf{x}}}$$

❖ Logistic regression



If we set the logistic regression threshold to 0.5, it means all predictions with a value bigger than 0 will be classified as 1 and all predictions with a value smaller than 0 will be classified as 0.

-10.26391683

❖ Logistic regression

Example

We performed a logistic regression classification on the weight-height dataset to predict the participants' gender using their weight and height.

Since we have two inputs, weight and height, the logistic regression hypothesis will be as follows:

$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$$

The logistic regression classifier will predict "Male" (i.e. 1) if:

$$\theta_0 + \theta_1 x_1 + \theta_2 x_2 \geq 0$$

-10.26391683

❖ Logistic regression

Example

The model coefficients were calculated as follows:

$$\theta^T = \begin{bmatrix} -10.26391683 \\ -1.76889234 \\ 23.66840149 \end{bmatrix}$$

Now if we want to predict the regression output for input (70,180) we have:

$$y = \begin{bmatrix} -10.26391683 \\ -1.76889234 \\ 23.66840149 \end{bmatrix} [1 \quad 70 \quad 180] = 4126.22588757$$

-10.26391683

❖ Logistic regression

Step 2: We need to find θ such that $h_\theta(x)$ is large when x belongs to the "1" class and small when x belongs to the "0" class.

For a set of training examples with binary labels $\{(x^{(i)}, y^{(i)}) : i = 1, \ldots, m\}$

the following cost function measures how well a given $h_\theta(x)$ does this

$$J(\theta) = -\sum_i \left(y^{(i)} \log(h_\theta(x^{(i)})) + (1 - y^{(i)}) \log(1 - h_\theta(x^{(i)})) \right)$$

-10.26391683

❖ Logistic regression

When $y^{(i)} = 1$ minimizing the cost function means we need to make $h_\theta(x^{(i)})$ large, and when $y^{(i)} = 0$ we want to make $1 - h_\theta(x^{(i)})$ large

This cost function is called a log loss or a binary cross entropy function

Therefore, to find optimum θ values, we need to minimise $J(\theta)$ as follows

$$\min_\theta J(\theta)$$

-10.26391683

❖ Logistic regression

- **Gradient Descent**

To minimize *J*, we can use the gradient descent method. The derivative of *J* will be given as follows:

$$\frac{\partial J(\theta)}{\partial \theta_j} = \sum_i x_j^{(i)} (h_\theta(x^{(i)}) - y^{(i)})$$

Therefore, θ will be updated in each iteration of the gradient descent method using the following equation

$$\theta_j^{k+1} = \theta_j^k - \eta \frac{\partial J(\theta)}{\partial \theta_j^k}$$

-10.26391683

❖ Multi class logistic regression

The logistic regression method was proposed for binary classification in which the target vector (label column) has only two classes.

However, three extensions to logistic regression are available to use logistic regression for multiclass classification in which the target vector has more than two classes.

These extensions include:

- One-vs-Rest (OvR) multiclass strategy
- One-vs-One (OvO) multiclass strategy
- Multinomial method

-10.26391683

❖ Multi class logistic regression

- **OvR**

When there are more than two classes in the target vector, the one-vs-rest strategy allows logistic regression to train a separate model for each class comparing with all the remaining classes. Therefore, this is also known as the One-vs-All (OvA) strategy

-10.26391683

❖ Multi class logistic regression

- **OvR**

This method creates one logistic regression model for each class against all other classes. If the target vector has four classes (e.g. Cat, Dog, Monkey, Bear), this strategy will create four separate models in the following way.

Model 1: Cat vs [Dog, Monkey, Bear]
Model 2: Dog vs [Cat, Monkey, Bear]
Model 3: Monkey vs [Cat, Dog, Bear]
Model 4: Bear vs [Cat, Dog, Monkey]

-10.26391683

❖ Multi class logistic regression

- **OvO**

When there are more than two classes in the target vector, the one-vs-one strategy allows logistic regression to train a separate model for each class against every individual remaining class.

If the target vector has n number of classes, this strategy will create n * (n-1) / 2 models.

-10.26391683

❖ Multi class logistic regression

- **OvO**

Let's say the classes are Cat, Dog, Monkey and Bear. This strategy will create six separate models in the following way.

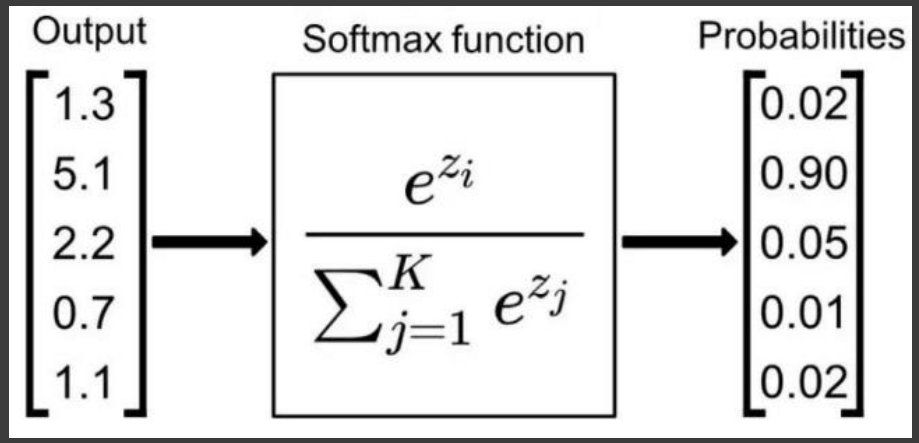
Model 1: Cat vs Dog
Model 2: Cat vs Monkey
Model 3: Cat vs Bear
Model 4: Dog vs Monkey
Model 5: Dog vs Bear
Model 6: Monkey vs Bear

-10.26391683

❖ Multi class logistic regression

- **Multinomial method**

In the multinomial method, the logistic function is replaced with the softmax function which calculates the probability value of each class over n different classes. The class with the highest probability will then be selected.



-10.26391683

❖ Multi class logistic regression

- **Multinomial method**

The objective function (loss function) will be a log loss function as follows:

$$logloss = -\frac{1}{N}\sum_{i}^{N}\sum_{j}^{M} y_{ij}\log(p_{ij})$$

- N is the number of rows
- M is the number of classes

This cost function is also called a multi-class cross-entropy function or a categorical cross-entropy function

-10.26391683

❖ Logistic regression: binary vs multinomial

- In a binary logistic regression, the module generates only one output which will be converted into a probability
- In a multinomial logistic regression with K classes, the model generates K output which will be converted into K probability values using the softmax function.

-10.26391683

❖ Classification using Neural network

- **Binary classification**

To use neural networks for binary classification we need to make the following modifications to the neural network structure:

- Use a binary cross-entropy function as the loss function
- Set the output layer activation function to a sigmoid function

-10.26391683

❖ Classification using Neural network

- **Multi-class classification**

To use neural networks for multi-class classification we need to make the following modifications to the neural network structure:

- One hot encode the target data used for training
- Use a categorical cross-entropy function as the loss function
- Set the output layer activation function to a softmax function

the following cost function measures how well a given line does this

-:

-10.26391683

❖ Python Examples

See the notebook ‘6CS012 lecture example’ for Python examples