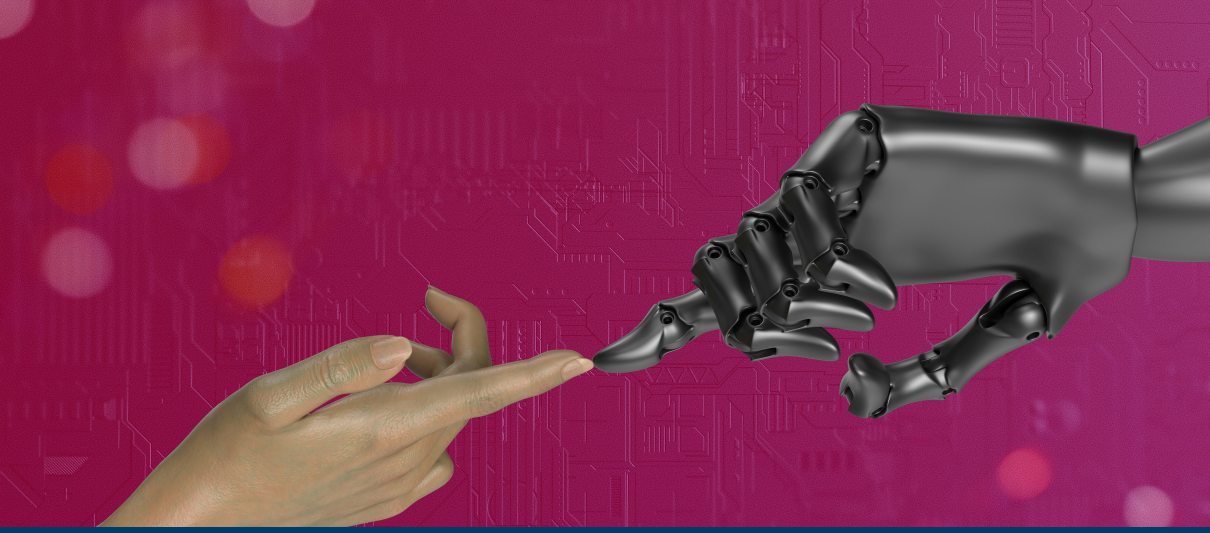


Hands-On Machine Learning

Dr. Derek Bridge | 2026/2027

UCC Coláiste na hOllscoile Corcaigh
University College Cork, Ireland

School of Computer Science and Information Technology
Scoil na Ríomheolaíochta agus na Teicneolaíochta Faisnéise

9. Logistic Regression

In this lecture, for binary classification we use the same student data that we used in a previous lecture. For multiclass classification, we use the Iris dataset. This widely-used dataset was created by the (in)famous statistician, Ronald Fisher (Fisher, R.A. "The use of multiple measurements in taxonomic problems", Annual Eugenics, 7, Part II, 179-188, 1936). The dataset is available all over the web now, especially the following dataset repository: https://archive.ics.uci.edu/ml/datasets/iris

	Regression	Classification
Decision Trees	Use MSE as the splitting criterion	Use, e.g., Gini as the splitting criterion
kNN	Take the mean of the neighbours' y-values	Take a majority-vote of the neighbours' y-values
Linear Models	Linear Regression, e.g. OLS regression which uses MSE as the loss function	Logistic Regression using cross-entropy as the loss function

Linear Regression

- In **Linear Regression** we want to find the line/plane/hyperplane that best fits the training examples — it 'goes through the middle of them'.

Logistic Regression

- Despite its name, **Logistic Regression** is a classifier, not a regressor.
- In Logistic Regression for binary classification, we want to find the line/ plane/ hyperplane that best *separates* training examples of different classes.
- Ideally, we want positive examples on one side and negatives on the other side. If we get this, we say that the dataset is **linearly separable**.

Revision

- Given an object $\boldsymbol{x}$, a classifier outputs a label, $\hat{y} \in \mathcal{C}$.
- Instead, a classifier could output a probability distribution over the labels $\mathcal{C}$.
- E.g. given an email $\boldsymbol{x}$, a spam filter might output $\langle 0.2, 0.8 \rangle$ meaning the probability that $\boldsymbol{x}$ is *ham* is 0.2, and the probability that it is *spam* is 0.8.
- The probabilities must sum to 1.
- In fact, a binary classifier can output just one probability, rather than two. Why?

Logistic Regression

- Given an example $\boldsymbol{x}$, Logistic Regression predicts the probability that $\boldsymbol{x}$ belongs to the positive class, $Prob(\hat{y} = 1 \mid \boldsymbol{x})$.

Logistic Regression

- Logistic Regression is a linear model, so we are trying to find values for b and $w_1, \ldots, w_n$:

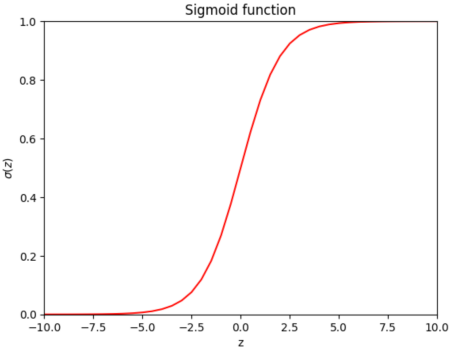
$$z = b + \boldsymbol{w}_1 \boldsymbol{x}_1 + \cdots + \boldsymbol{w}_n \boldsymbol{x}_n$$

- But we need it to produce a probability.
- To 'squash' z to $[0, 1]$, we use the **sigmoid function** (also called the **logistic function** or the **logit**):

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

- Putting it all together:

$$Prob(\hat{y} = 1 \mid \boldsymbol{x}) = \sigma(b + \boldsymbol{w}_1 \boldsymbol{x}_1 + \cdots + \boldsymbol{w}_n \boldsymbol{x}_n)$$

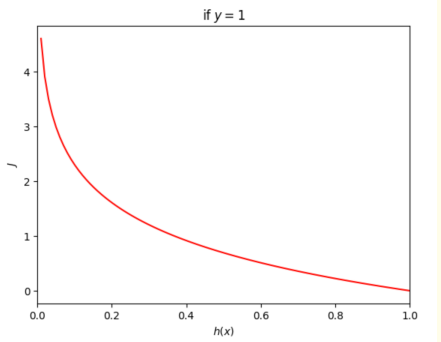


The Loss Function for Logistic Regression

- During training, we must find values for the parameters, $b, \boldsymbol{w}_1, \ldots, \boldsymbol{w}_n$.
- Recall that we refer to each choice as a **hypothesis**, h.
- We need a **loss function** to score the hypotheses — we pick the hypothesis with lowest loss.
- We need a loss function, which penalizes a hypothesis if
  - it outputs high probabilities for negative examples; and
  - it outputs low probabilities for positive examples.
- The loss function that we will use is called **binary cross-entropy** and is based on logarithms. (It is also called the **log loss function**.)

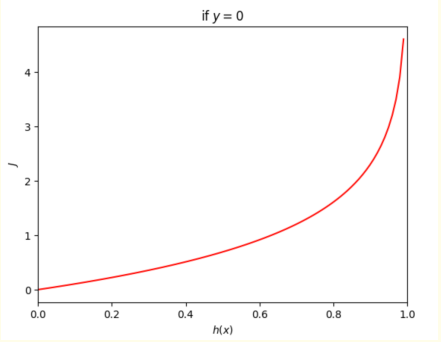
Positive Examples

- Consider a single training example $\boldsymbol{x}$ whose actual class $y = 1$.
- $-\log(h(\boldsymbol{x}))$ is a useful loss function:



Negative Examples

- Consider a single training example $\boldsymbol{x}$ whose actual class $y = 0$.
- $-\log(1 - h(\boldsymbol{x}))$ is a useful loss function:



Binary Cross-Entropy

- In summary, for a given example $\boldsymbol{x}$, we propose the following as the loss function

$$\begin{cases} -\log(h(\boldsymbol{x})) & \text{if } y = 1 \\ -\log(1 - h(\boldsymbol{x})) & \text{if } y = 0 \end{cases}$$

  Or, equivalently:

$$-y \log(h(\boldsymbol{x})) + -(1 - y) \log(1 - h(\boldsymbol{x}))$$

  Or

$$-[y \log(h(\boldsymbol{x})) + (1 - y) \log(1 - h(\boldsymbol{x}))]$$

- The overall loss function, J, is simply the average of this over all the training examples:

$$J(\boldsymbol{X}, \boldsymbol{y}, h) = -\frac{1}{m} \sum_{\boldsymbol{x} \in \boldsymbol{X}, y \in \boldsymbol{y}} [y \log(h(\boldsymbol{x})) + (1 - y) \log(1 - h(\boldsymbol{x}))]$$

Finding the Hypothesis that Minimizes Binary Cross-Entropy

- Happily, this loss function is convex. What does that mean?
- But there is no equivalent to the Normal Equation.
- We must use Gradient Descent.
- Since we are using Gradient Descent, we must remember to scale the data.

Stochastic Gradient Descent

initialize $\boldsymbol{w}$ randomly;

for *each epoch* **do**
  shuffle the training set;
  for *each example* $\boldsymbol{x}$ **do**
    $\boldsymbol{w} \leftarrow \boldsymbol{w} - \alpha \boldsymbol{x}(\sigma(\boldsymbol{w}\boldsymbol{x}) - y)$;
  end
end

Types of Classification

- **Binary Classification**: there are just two classes, i.e. $|\mathcal{C}| = 2$, e.g. fail/pass, ham/spam, benign/malignant.
- **Multiclass Classification**: there are more than two classes, i.e. $|\mathcal{C}| > 2$
  - E.g. a post to a forum or discussion board can be a question, an answer, a clarification or an irrelevance.
  - E.g. an iris can be Iris setosa, Iris versicolor or Iris virginica

Even More Types of Classification

- **Multilabel Classification**: the classifier can assign $\boldsymbol{x}$ to more than one class.
  - The classifier outputs a *set* of labels, $\hat{y} \subseteq \mathcal{C}$.
  - E.g. consider a movie classifier where the classes are genres, e.g. $\mathcal{C} = \{comedy, action, horror, musical, romance\}$ The classifier's output for *The Blues Brothers* should be $\{comedy, action, musical\}$.

  Do *not* confuse this with **multiclass classification**.
- **Ordered classification**: there is an *ordering* defined on the classes.
  - The ordering matters in measuring the performance of the classifier.
  - E.g. consider a classifier that predicts a student's degree class, i.e. $\mathcal{C} = \{Ordinary, 3rd, 2ii, 2i, 1st\}$
    - Suppose for student $\boldsymbol{x}$, the actual class $y = 1st$
    - One classifier predicts $\hat{y} = 2ii$
    - Another classifier predicts $\hat{y} = 2i$
    - Which classifier has done better?

Multiclass Classifiers

- Decision Trees can be used for multiclass classification without change.
- kNN classifiers also.
- But Logistic Regression is for binary classification.

Multinomial Logistic Regression

- **Multinomial Logistic Regression** is for multiclass classification.
- Instead of learning one set of parameters, it learns one per class — in other words, it is learning a line/ plane/ hyperplane for each class.
- Instead of outputting one probability, it outputs $|\mathcal{C}|$ probabilities — one per class.
- 'Squashing' is more complicated because not only must each output be squashed to $[0, 1]$, they must also sum to 1. So we do not use the sigmoid function. We use the **softmax function**.
- In place of binary cross-entropy, we use a generalization called **categorical cross-entropy**.

For those who want all the maths

Multinomial Logistic Regression

- Multinomial Logistic Regression learns a set of parameters $\boldsymbol{w}_c$ for each class $c \in \mathcal{C}$
- Then, the probability that $\boldsymbol{x}$ belongs to class c will use $\boldsymbol{w}_c$ with the **softmax** function:

$$Prob(\hat{y} = c \mid \boldsymbol{x}) = \sigma(\boldsymbol{w}_c \boldsymbol{x}) = \frac{e^{\boldsymbol{w}_c \boldsymbol{x}}}{\sum_{\kappa \in \mathcal{C}} e^{\boldsymbol{w}_\kappa \boldsymbol{x}}}$$

Categorical Cross-Entropy

- The loss function for Multinomial Logistic Regression generalises binary cross-entropy:

$$J(\boldsymbol{X}, \boldsymbol{y}, h) = \frac{1}{m} \sum_{\boldsymbol{x} \in \boldsymbol{X}, y \in \boldsymbol{y}} \sum_{\kappa \in \mathcal{C}} int(y = \kappa) log(\sigma(\boldsymbol{w}_\kappa \boldsymbol{x}))$$