Artificial Intelligence I 2023/2024

Week 4 Tutorial and Additional Exercises

Logistic Regression & kNN

School of Computer Science

February 13, 2024

Part One: Logistic Regression...

In tutorial part one, we will be covering

- Univariate and multivariate logistic regression.
- Geometric concepts.
- Advanced theoretical exercises.

Univariate logistic regression

Recall the formal statement of *univariate logistic regression*:

- Given a training set $\{(x^{(1)}, y^{(1)}), \ldots, (x^{(n)}, y^{(n)})\}$, where $y^{(i)} \in \{0, 1\}$ for all $i = 1, \ldots, n$, train weights w_0, w_1 that minimise a loss function.
- Given this training set, and weights w_0, w_1, the *logistic loss* (or *cross-entropy loss*) function is given as

$$g(w_0, w_1) = -\frac{1}{n} \sum_{i=1}^{n} \Big(y^{(i)} \ln(\sigma(w_0 + w_1 x^{(i)})) + (1 - y^{(i)}) \ln(1 - \sigma(w_0 + w_1 x^{(i)})) \Big)$$

- where $\sigma(x) = \dfrac{1}{1 + e^{-x}}$ is the *sigmoid* function.

Multivariate logistic regression

Recall the formal statement of *multivariate logistic regression*:

- Given a training set $\{(\mathbf{x}^{(1)}, y^{(1)}), \ldots, (\mathbf{x}^{(n)}, y^{(n)})\}$, where $y^{(i)} \in \{0, 1\}$ for all $i = 1, \ldots, n$, train a weight vector $\mathbf{w}$ that minimizes a loss function.
- If we have d variables, then for all $i = 1, \ldots, n$, we write

$$\mathbf{x}^{(i)} = (1, x_1^{(i)}, x_2^{(i)}, \ldots, x_d^{(i)}) \text{ and } \mathbf{w} = (w_0, w_1, w_2, \ldots, w_d).$$

- Given this training set and a weight vector $\mathbf{w}$, the *logistic loss* (or cross-entropy loss) function is given as

$$g(\mathbf{w}) = -\frac{1}{n} \sum_{i=1}^{n} \Big(y^{(i)} \ln(\sigma(\mathbf{w}^T \mathbf{x}^{(i)})) + (1 - y^{(i)}) \ln(1 - \sigma(\mathbf{w}^T \mathbf{x}^{(i)})) \Big) .$$

Exercise 1

- Consider a logistic regression model with 2 variables that given an instance $\mathbf{x} = (x_1, x_2)$ and weights w_0, w_1, w_2, it predicts the label of $\mathbf{x}$ to be

$$\hat{y} = \begin{cases} 1 \text{ if } w_0 + w_1 x_1 + w_2 x_2 > 0 \\ 0 \text{ if } w_0 + w_1 x_1 + w_2 x_2 < 0 \end{cases}$$

- For each of the following cases, draw the decision boundary in the $x_1 x_2$-plane. This is the line where $w_0 + w_1 x_1 + w_2 x_2 = 0$. Also draw the labels corresponding to the two resulting areas.
  1. $w_0 = 1,\ w_1 = 1,\ w_2 = 2$.
  2. $w_0 = 0,\ w_1 = -3,\ w_2 = 1$.
  3. $w_0 = -2,\ w_1 = 4,\ w_2 = 0$.
  4. $w_0 = -2,\ w_1 = 0,\ w_2 = -1$.

Exercise 2

- Logistic regression creates a decision boundary (e.g. a line for two variables) and predicts the label of an instance according to which side of the boundary it falls into.
- Assume each instance has two variables (x_1, x_2), and a label $y \in \{0, 1\}$. Design two training sets that logistic regression can separate with a line, and two training sets that logistic regression cannot separate with a line.
  For each point, write the values of its two variables and its label; or plot the points of the training set in the x_1x_2-plane to show whether a line can separate all instances.

Exercise 3

- This exercise studies the power of a linear decision boundary, as the maximum number of instances it can separate.
- Reconsider the case where each instance has two variables (x_1, x_2) and a label of either 0 or 1.
- Can you plot three instances in the x_1x_2-plane, not all three in the same line, such that no line can separate the two labels? You can freely choose the label of each instance.
- Can you plot four instances in the x_1x_2-plane, no three in the same line, such that no line can separate the two labels? You can freely choose the label of each instance.
- In learning theory, this notion is called the *VC-dimension* (out of the scope of this module).

Advanced Material

(OPTIONAL) Advanced Exercise 1

- Consider the *sigmoid* function

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

- (1) Show that σ is an increasing function and only takes values in $[0,1]$.
- (2) Can σ take on the values of 0 or 1 for some x?
- Hint: To show that σ is increasing, show that $\sigma'(x) > 0$, for all x. To show that σ takes values in $[0,1]$, find the limits

$$\lim_{x\to-\infty} \sigma(x) \quad \text{and} \quad \lim_{x\to\infty} \sigma(x).$$

- Hint: To find whether σ takes on the values of 0 or 1, solve

$$\sigma(x) = 0 \quad \text{and} \quad \sigma(x) = 1.$$

Advanced Exercise 2

- Let $(\mathbf{x}, y)$ be a data point and $\mathbf{w}$ be the weight vector to be optimised in a multivariate logistic regression model with d variables. Assume that $\mathbf{x}$ and $\mathbf{w}$ are of the form[1]

$$\mathbf{x} = (x_0, x_1, \ldots, x_d) \text{ and } \mathbf{w} = (w_0, w_1, \ldots, w_d).$$

- Let $\sigma(x) = \frac{1}{1+e^{-x}}$ and g be the logistic loss function

$$g(\mathbf{w}) = -\left(y \ln(\sigma(\mathbf{w}^T\mathbf{x})) + (1-y)\ln(1-\sigma(\mathbf{w}^T\mathbf{x}))\right).$$

- Use the derivative rules to prove that

$$\nabla g(\mathbf{w}) = -(y - \sigma(\mathbf{w}^T\mathbf{x}))\mathbf{x}.$$

- Hint: $\frac{\partial \sigma}{\partial w_i}(\mathbf{w}^T\mathbf{x}) = \sigma(\mathbf{w}^T\mathbf{x})(1-\sigma(\mathbf{w}^T\mathbf{x}))x_i$, $i = 0, \ldots, d$.

[1] We usually take $x_0 = 1$, but we leave it as x_0 here.

Part Two: kNN...

In tutorial part 2, we will be covering

- Distance metrics.
- Normalisation.
- *k*-nearest neighbours.
- Advanced theoretical exercises.

Parametric and non-parametric models

- *Parametric models* are learning models that summarise data with a set of parameters.
- Linear and logistic regression are examples of parametric models.
- *Non-parametric models* are learning models that do not assume any parameters.
- kNN is a non-parametric model.

Distance metrics

- A *distance metric* is a way to quantify the similarity or dissimilarity between instances.
- There are many available distance metrics. We need to choose the one that best fits the problem at hand.
- A distance metric takes two vectors as inputs and outputs a non-negative number.
- Different notions of distance metrics are used for vectors of numerical variables and for vectors of categorical variables.

Distance metrics (continued)

- For numerical variables, we will use the *Minkowski distance*.
- Given a number $p \geq 1$ and two vectors with d numerical variables

$$\mathbf{x}^{(1)} = (x_1^{(1)}, \ldots, x_d^{(1)}) \quad \text{and} \quad \mathbf{x}^{(2)} = (x_1^{(2)}, \ldots, x_d^{(2)})$$

their Minkowski distance (or L^p-norm) is defined as

$$L^p(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}) = \sqrt[p]{\sum_{j=1}^{d} \left| x_j^{(1)} - x_j^{(2)} \right|^p}.$$

- For $p = 2$ we obtain the *Euclidean distance*.
- For $p = 1$ we obtain the *Manhattan distance*.

Exercise 1

- Consider the following vectors with 3 numerical variables.

$$\mathbf{x}^{(1)} = \begin{bmatrix} 0 \\ 3 \\ -1 \end{bmatrix}, \mathbf{x}^{(2)} = \begin{bmatrix} -2 \\ 3 \\ -1 \end{bmatrix}, \mathbf{x}^{(3)} = \begin{bmatrix} 1 \\ -1 \\ 1 \end{bmatrix}, \mathbf{x}^{(4)} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}.$$

- Compute the Euclidean and Manhattan distance matrices for these vectors.
- Hint: You need to compute 6 distances on total.

Distance metrics (continued)

- For categorical variables, we will use the *Hamming distance*.
- Given two vectors with d categorical variables

$$\mathbf{x}^{(1)} = (x_1^{(1)}, \ldots, x_d^{(1)}) \quad \text{and} \quad \mathbf{x}^{(2)} = (x_1^{(2)}, \ldots, x_d^{(2)}),$$

their Hamming distance is defined as

$$H(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}) = \sum_{j=1}^{d} \mathbf{1}(x_j^{(1)} \neq x_j^{(2)})$$

where $\mathbf{1}$ is the indicator function. For all $j = 1, \ldots, d$, we have

$$\mathbf{1}(x_j^{(1)} \neq x_j^{(2)}) = \begin{cases} 1 \text{ if } x_j^{(1)} \neq x_j^{(2)} \\ 0 \text{ if } x_j^{(1)} = x_j^{(2)}. \end{cases}$$

Exercise 2

- Consider the following vectors with 1 ordinal variable (the first attribute) and 3 categorical variables (the remaining 3 attributes). For the ordinal attribute, simply transform it into numerical values: yes $\rightarrow$ 1 and no $\rightarrow$ 0. For the categorical ones, use the hamming distance.

$$\mathbf{x}^{(1)} = \begin{bmatrix} \textit{yes} \\ \textit{red} \\ \textit{FR} \\ \triangle \end{bmatrix}, \mathbf{x}^{(2)} = \begin{bmatrix} \textit{yes} \\ \textit{blue} \\ \textit{FR} \\ \square \end{bmatrix}, \mathbf{x}^{(3)} = \begin{bmatrix} \textit{no} \\ \textit{green} \\ \textit{UK} \\ \bigcirc \end{bmatrix}, \mathbf{x}^{(4)} = \begin{bmatrix} \textit{yes} \\ \textit{red} \\ \textit{DE} \\ \triangle \end{bmatrix}.$$

- Find the distance matrix for these vectors.
- Hint: You need to compute 6 distances on total. Each distance can be at most 4.

Normalisation

- *Normalisation* is used to restrict numerical variables in $[0,1]$.
- Given a set of n vectors $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}$, with d numerical variables, for all $j = 1, \ldots, d$, we write

$$\min_j = \min\{x_j^{(1)}, \ldots, x_j^{(n)}\} \text{ and } \max_j = \max\{x_j^{(1)}, \ldots, x_j^{(n)}\}.$$

- Then, the j-th variable of the i-th vector is normalised as

$$\mathit{normalise}(x_j^{(i)}) = \frac{x_j^{(i)} - \min_j}{\max_j - \min_j}.$$

- We calculate the above formula for all $i = 1, \ldots, n$ and for all $j = 1, \ldots, d$ and normalise all variables in all vectors.

Exercise 3

- Consider the following vectors with 3 numerical variables.

$$\mathbf{x}^{(1)} = \begin{bmatrix} -2 \\ 3 \\ 300 \end{bmatrix}, \mathbf{x}^{(2)} = \begin{bmatrix} 2 \\ 1 \\ -100 \end{bmatrix}, \mathbf{x}^{(3)} = \begin{bmatrix} 0 \\ 2 \\ 100 \end{bmatrix}, \mathbf{x}^{(4)} = \begin{bmatrix} 1 \\ 2 \\ -200 \end{bmatrix}.$$

- Normalise all variables in all vectors, using the methodology we just described.
- Hint: First compute $\min_j$ and $\max_j$ for all $j = 1, 2, 3$. Then use the normalisation formula.

Mixed distance

- When vectors have both numerical and categorical variables, we need to use some combination of distance metrics.
- One way is to use the *mixed distance*.
- Given a number $p \geq 1$ and two vectors with d variables

$$\mathbf{x}^{(1)} = (x_1^{(1)}, \ldots, x_d^{(1)}) \quad \text{and} \quad \mathbf{x}^{(2)} = (x_1^{(2)}, \ldots, x_d^{(2)}),$$

their mixed distance is defined as

$$D^p(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}) = \sqrt[p]{\sum_{j=1}^{d} |\bar{x}_j|^p}$$

where

$$\bar{x}_j = \begin{cases} normalise(x_j^{(1)}) - normalise(x_j^{(2)}), & \text{if } j \text{ is numerical} \\ \mathbf{1}(x_j^{(1)} \neq x_j^{(2)}), & \text{if } j \text{ is categorical.} \end{cases}$$

k-nearest neighbours

- *k-nearest neighbours* (k-NN) is one of the most popular classification algorithms.
- Given a labeled training set, we predict the labels of future instances depending on their distance from the labeled ones.
- k determines how many of the closest labeled instances to consider. We need to choose its value ourselves.
- Different notions of distance can be used, depending on the types of variables.
- In our examples we will use *mixed Euclidean distance*, which is mixed distance for $p = 2$.

Exercise 4

- Consider the following data set.

	gen	age	bmi	city	ill
$\mathbf{x}^{(1)}$	male	33	28.8	Bristol	no
$\mathbf{x}^{(2)}$	female	45	23.8	London	no
$\mathbf{x}^{(3)}$	female	68	21.3	Edinburgh	yes
$\mathbf{x}^{(4)}$	male	21	22.6	London	yes
$\mathbf{x}^{(5)}$	male	71	18.3	Birmingham	no
$\mathbf{x}^{(6)}$	female	27	28	Birmingham	yes
$\mathbf{x}^{(new)}$	female	26	20	Birmingham	?

- Use k-NN to find the missing value of $\mathbf{x}^{(new)}$.
- Use $k = 3$ and the mixed Euclidean distance. Use the majority vote to determine the label for $\mathbf{x}^{(new)}$.

Advanced Material

Weighted k-NN

- In k-NN, ties can occur that need to be broken somehow.
- One way is to use a version of k-NN called *weighted k-NN*.
- First, we convert each label as $\{\text{no}, \text{yes}\} \rightarrow \{0, 1\}$.
- Then, each point $\mathbf{x}^{(i)}$ is given a weight w_i using a function called *kernel function*. We then calculate a weighted sum:

$$S = \frac{1}{\sum_{i \in \mathcal{N}_k} w_i} \sum_{i \in \mathcal{N}_k} w_i y^{(i)}$$

  where $\mathcal{N}_k$ is the set of indices of the k closest points to $\mathbf{x}^{(new)}$.
- The label $y^{(new)}$ is then predicted using the formula:

$$\hat{y}^{(new)} = \begin{cases} \text{yes, if } S > 0.5 \\ \text{no, if } S \leq 0.5. \end{cases}$$

- The most straightforward kernel function is the inverse of the mixed Euclidean distance, that is, $w_i = 1/Dist(\mathbf{x}^{(new)}, \mathbf{x}^{(i)})$.

Advanced Exercise 1

- Reconsider this distance table with $y^{(i)}$ converted to $\{0, 1\}$.

	gen	age	bmi	city	$D^2(\cdot)$	$y^{(i)}$
$Dist(\mathbf{x}^{(new)}, \mathbf{x}^{(1)})$	1	0.0196	0.70	1	1.65	0
$Dist(\mathbf{x}^{(new)}, \mathbf{x}^{(2)})$	0	0.1444	0.13	1	1.13	0
$Dist(\mathbf{x}^{(new)}, \mathbf{x}^{(3)})$	0	0.7056	0.015	1	1.31	1
$Dist(\mathbf{x}^{(new)}, \mathbf{x}^{(4)})$	1	0.01	0.06	1	1.44	1
$Dist(\mathbf{x}^{(new)}, \mathbf{x}^{(5)})$	1	0.81	0.026	0	1.36	0
$Dist(\mathbf{x}^{(new)}, \mathbf{x}^{(6)})$	0	0.0004	0.58	0	0.76	1

- What will weighted k-NN predict for $y^{(new)}$ with $k = 4$ and the mixed Euclidean distance as the kernel function?
- Hint: Calculate

$$S = \frac{1}{\sum_{i \in \mathcal{N}_k} 1/Dist(\mathbf{x}^{(new)}, \mathbf{x}^{(i)})} \sum_{i \in \mathcal{N}_k} \frac{y^{(i)}}{Dist(\mathbf{x}^{(new)}, \mathbf{x}^{(i)})}$$ where

$k = 4$ and $\mathcal{N}_4 = \{2, 3, 5, 6\}$. Predict yes if $S > 0.5$ or no if $S \leq 0.5$.

(OPTIONAL) Advanced Exercise 2

Definition 1 (Distance metric)

A function $f : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ can be used to define a *distance metric*, if and only if, for all vectors $\mathbf{x}, \mathbf{y}, \mathbf{z} \in \mathcal{X}$, the following hold:

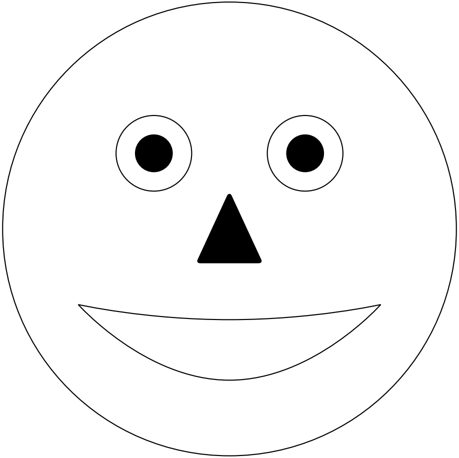
1. $f(\mathbf{x}, \mathbf{y}) = 0$, if and only if, $\mathbf{x} = \mathbf{y}$;
2. $f(\mathbf{x}, \mathbf{y}) = f(\mathbf{y}, \mathbf{x})$; and
3. $f(\mathbf{x}, \mathbf{z}) \leq f(\mathbf{x}, \mathbf{y}) + f(\mathbf{y}, \mathbf{z})$.

- Show that the Minkowski distance L^p, (for any $p \geq 1$) and the Hamming distance H, are distance metrics.
- Hint: Use *Minkowski's inequality*: for all $a_1, a_2 \ldots, a_d \in \mathbb{R}$ and $b_1, b_2 \ldots, b_d \in \mathbb{R}$ and $p \geq 1$, we have

$$\sqrt[p]{\sum_{j=1}^{d} |a_j + b_j|^p} \leq \sqrt[p]{\sum_{j=1}^{d} |a_j|^p} + \sqrt[p]{\sum_{j=1}^{d} |b_j|^p}.$$

Any questions?

Until the next time...



Thank you for your attention!