Artificial Intelligence and Machine Learning

Exercises – Logistic Regression

Question 1 (Why you should not use linear regression for classification)

The following dataset contains examples of *planets* and *dwarf planets*. Each celestial body in the dataset is described only by its radius, i. e. the distance from the core to the surface, in millions of meters. Based on this feature we want to learn a classification model which predicts if the object is either a *planet* or a *dwarf planet*.

Table 1 as well as figure 1 introduce the dataset consisting of $n = 6$ training instances:

Row	Object	Radius ($\times 10^6$ m)	Label	Label encoded
1	Ceres	1.0	dwarf planet	0
2	Eris	2.3	dwarf planet	0
3	Pluto	2.4	dwarf planet	0
4	Mercury	4.9	planet	1
5	Earth	12.8	planet	1
6	Jupiter	143.0	planet	1

Table 1: Dataset of planets and dwarf planets.

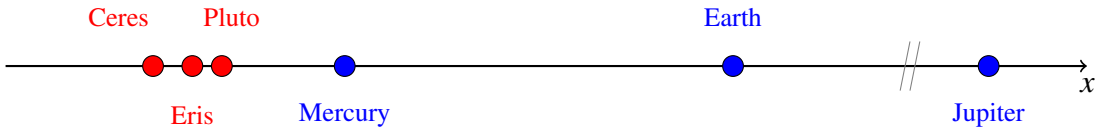

Figure 1: Visualization of the planet dataset (not drawn to scale).

Please work through the following tasks:

1. As a baseline you decide to use a linear regression model. Compute the decision boundary using the normal equations for linear regression. The last column in table 1 contains the encoded target labels to be used. Apply the threshold $\rho := 0.5$, i. e. predict the positive class, *planet*, if the model output is greater or equal to 0.5, and the negative class, *dwarf planet*, otherwise. What problem do you observe?

2. Train a logistic regression classifier on the same data. To achieve this, implement the batch gradient descent algorithm for logistic regression (e. g. in Python). Use the learning rate $\alpha := 0.075$ and initialize the algorithm at the point $\boldsymbol{\theta}^0 := \begin{pmatrix} 0 & 0 \end{pmatrix}^\top$.

   - What are the model parameters $\boldsymbol{\theta}^{20}$ after 20 iterations of gradient descent?

   - Feed the training data into your trained classifier. Again using the threshold $\rho := 0.5$: What are the predictions of the model?

- Compute the decision boundary generated by your model! How does logistic regression perform compared to linear regression?
- Is your logistic regression model Bayes optimal?

Question 2 (Derivative of the sigmoid function)

The derivative of the sigmoid function

$$\sigma(z) := \frac{1}{1 + \exp(-z)}$$

is crucial for the training of a logistic regression classifier. In the lecture slides we have seen that the derivative is given by

$$\frac{\mathrm{d}}{\mathrm{d}z}\sigma(z) = \sigma(z) \cdot (1 - \sigma(z)).$$

Verify that this is the correct derivative of the sigmoid function.

Question 3 (Stochastic gradient descent)

Let the training example

$$\boldsymbol{x} := \begin{pmatrix} -3 & 1 & -1 & 1 & 0 \end{pmatrix}^\top, \qquad y := 1$$

be given. Perform one iteration of (stochastic) gradient descent starting with the initial parameters $\boldsymbol{\theta}^0 := \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \end{pmatrix}^\top$. Use the learning rate $\alpha := 0.2$. What are advantages and disadvantages of stochastic gradient descent? What other types of gradient descent do you know? Briefly explain the concepts.

Question 4 (Logistic regression with basis functions)

Write down the model function $h_{\boldsymbol{\theta}}(\varphi(\boldsymbol{x}))$ of the model depicted in the following figure 2:

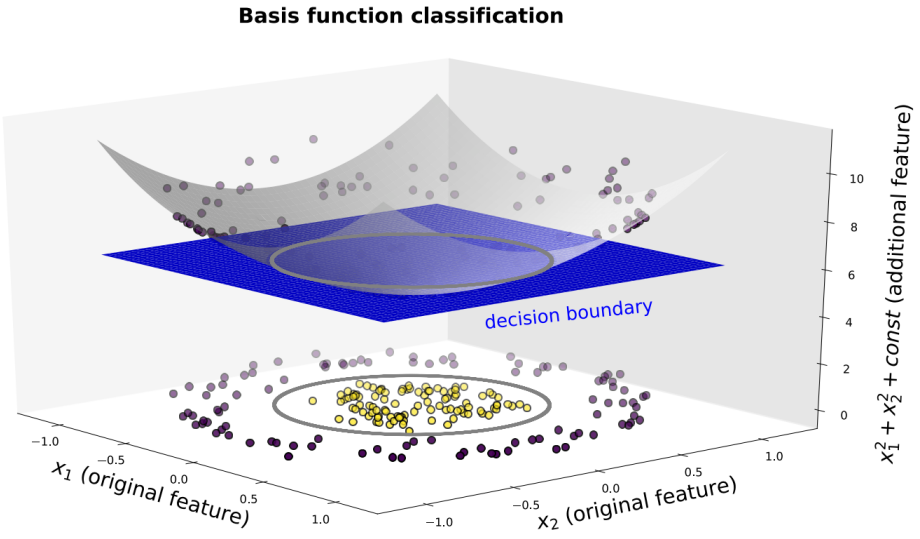

Figure 2: Logistic regression with polynomial basis functions.

Question 5 (Multi-class classification with logistic regression) ⊛

You want to automatically detect handwritten digits (10 possible classes). You decide to train a logistic regression classifier. You choose the One-vs-One approach, since logistic regression is a binary classification model.

How many (binary) classifiers do you have to train to complete the task?

Question 6 (One-vs-One and One-vs-Rest)

For different values of K (number of classes), compare the number of base classifiers to be trained when using the One-vs-One and One-vs-Rest approach, respectively. Which method is more expensive for large K?

Question 7 (Hinge loss classifier) ⊛

In this exercise you will derive the **hinge loss classifier** which is an alternative to logistic regression. Again we consider a labeled dataset

$$\mathcal{D} := \{(\boldsymbol{x}^1, y_1), \ldots, (\boldsymbol{x}^N, y_N)\}$$

consisting of N training examples. We use the symbols $\boldsymbol{x}^n := \begin{pmatrix} 1 & x_1 & \ldots & x_M \end{pmatrix}^\top \in \mathbb{R}^{M+1}$ to denote the feature vector of the n-th training example, and $y_n \in \{-1, +1\}$ for the respective label. *Please note that the negative class is represented by -1 instead of 0.*

Model function: We turn the linear regression model into a classifier by plugging the linear model into the sign function which returns $+1$ if the input is non-negative, and -1 otherwise:

$$\text{sign}(z) := \begin{cases} +1 & \text{if } z \geq 0 \\ -1 & \text{if } z < 0. \end{cases}$$

The model function of the hinge loss classifier is then given by

$$h_{\boldsymbol{\theta}}(\boldsymbol{x}) := \text{sign}(\boldsymbol{\theta}^\top \boldsymbol{x}). \tag{1}$$

Loss function: The model is trained using the **hinge loss**

$$\ell^{\text{Hinge}} := \max\{0, 1 - y \cdot \boldsymbol{\theta}^\top \boldsymbol{x}\} = \begin{cases} 1 - y \cdot \boldsymbol{\theta}^\top \boldsymbol{x} & \text{if } y \cdot \boldsymbol{\theta}^\top \boldsymbol{x} < 1 \\ 0 & \text{if } y \cdot \boldsymbol{\theta}^\top \boldsymbol{x} \geq 1. \end{cases} \tag{2}$$

Figure 3 depicts a plot of the hinge loss function. Minimizing the hinge loss leads to a **maximum margin classifier**. We can see this in the plot below: Suppose we have a training example $\boldsymbol{x}$ for which $\boldsymbol{\theta}^\top \boldsymbol{x}$ is between 0 and 1. Although classified correctly, i. e. y and $\boldsymbol{\theta}^\top \boldsymbol{x}$ have the same sign, the hinge loss still assigns a small error to that example.

Please work through the following tasks:

1. Briefly explain what a maximum margin classifier is. What are the advantages compared to logistic regression?

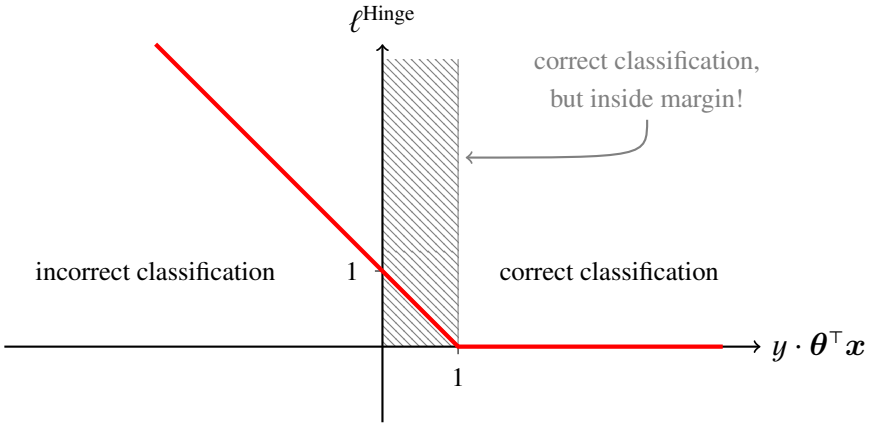

Figure 3: Visualization of the hinge loss function.

2. Compute the partial derivative of ℓ^{Hinge} defined in (2) with respect to the m-th model parameter θ_m and write down the gradient $\nabla_{\boldsymbol{\theta}} \ell^{\text{Hinge}}$. **Hint:** The hinge loss function (2) is not differentiable for $y \cdot \boldsymbol{\theta}^\top \boldsymbol{x} = 1$. Set the derivative to 0 in this case *(subgradient)*.

3. The listing below generates a small dataset. Train a hinge loss classifier on the data and plot the decision boundary.

4. Compare the decision boundary of your hinge loss classifier with those of linear regression (threshold $\rho := 0.5$) and logistic regression (threshold $\rho := 0.5$). What do you observe?

```
import numpy as np

X = np.asarray([
    [3.00, 1.00], [3.20, 2.20], [3.15, 4.80],
    [3.35, 1.20], [3.05, 3.50], [3.55, 2.85],
    [1.50, 2.25], [2.88, 2.18], [1.95, 4.00],
    [3.01, 2.95], [2.85, 3.01], [5.85, 2.20],
    [4.19, 4.00], [5.15, 3.50], [5.07, 2.89],
    [4.87, 3.54], [4.44, 3.78], [4.48, 3.94],
    [5.51, 3.80],
    # outliers
    [9.00, 5.00], [9.50, 5.25], [9.25, 5.50],
    [9.75, 5.75], [9.90, 4.80]
])

y = np.asarray([
    0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1,
    # outliers
    1, 1, 1, 1, 1
])
```

Question 8 (Derivative of the softmax function)

Compute the partial derivative of the k-th component function of the softmax function $\boldsymbol{\zeta}$ with respect to the j-th input z_j. The softmax function is defined by

$$\boldsymbol{\zeta} : \mathbb{R}^K \to \mathbb{R}^K, \qquad \boldsymbol{z} \mapsto \boldsymbol{\zeta}(\boldsymbol{z}), \qquad \zeta_k(\boldsymbol{z}) := \frac{e^{z_k}}{\sum_{n=1}^{K} e^{z_n}}.$$

Using this result, write down the Jacobian matrix $\frac{\partial \boldsymbol{\zeta}}{\partial \boldsymbol{z}} \in \mathbb{R}^{K \times K}$ of the softmax function.

Hint: Consider $\frac{\partial \zeta_k(\boldsymbol{z})}{\partial z_j}$ and examine the cases $k = j$ and $k \neq j$ separately.