Machine Learning

Group 16 - Assigment 2

Luca Pfrang	5332329
Tran Luong Bang	5365615
Louis Ngo	4506205
Vu Thi Hoang Anh	5367008
Tidiane Ndir	5364532

Freiburg, November 05 2021

1. LDA general

1. **What are the assumptions made by the LDA algorithm regarding the data distribution of the different classes?**

   - The data distributions of both classes are normally distributed
   - The covariance matrices of the classes are equal

2. **Could you mention a condition/scenario that would facilitate the application of LDA? How about a condition that makes it more difficult?**

   LDA performs well when the covariance of all the classes are similar and doesn't perform well when variances are very different

3. **What would you say are some of the main differences between the LDA and the KNN algorithm?**

   - LDA takes into account the whole dataset when building the model, whereas KNN only considers local neighbors.
   - LDA makes some assumptions about the distribution of the dataset, whereas KNN has no assumption.
   - LDA gives us the weight vectors, i:e tell us which variables are more important to the output.
   - LDA is not able to make reasonable predictions when the decision boundary is non-linear, as opposed to KNN.

2. LDA calculation

1. **Calculate mean m_k, and variance matrices S_k and S_w.**

   m_{NASA} = [4.8 14.5] $S_{NASA} = \begin{bmatrix} 0.35 & 49.33 \\ 1.98 & 31.5 \end{bmatrix}$

   m_{ALDI} = [7.5 49.33] $S_{ALDI} = \begin{bmatrix} 2.18 & 2.38 \\ 2.38 & 20.67 \end{bmatrix}$ $S_W = \begin{bmatrix} 1.264 & 2.18 \\ 2.18 & 25.88 \end{bmatrix}$

2. **Calculate w and b using the analytical form.**

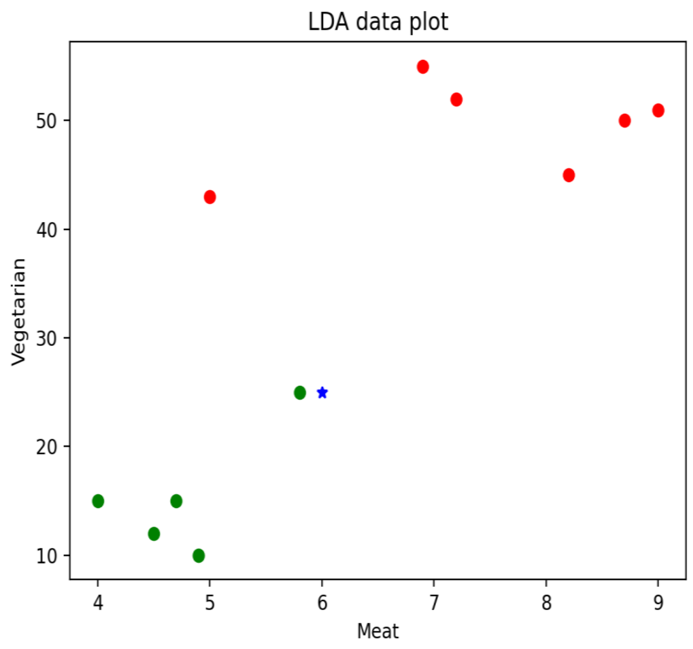
   w = [-0.2164 1.364],   b = -42.2

3. **Use your estimations to guess the logo for customer 13: Meat 6, Veg. 25**
4.

$$\hat{y} = w^T x + b = -9.4$$

Since $\hat{y} < 0$, we predict that customer [6, 25] will have logo NASA

5. **Sketch the data and the query point.**



3. Ordinary Least Squares Regression

1. **In its simplest form, what kind of interactions can the OLSR algorithm model? Can you mention an interaction that the algorithm cannot model well? How can the algorithm perform better in the latter situation?**

   When the output (regressand) is linearly dependent on the input variables (regressors). For example: housing price vs area of the house.
   When there is no linear relationship, OLSR performs poorly. For example: housing price vs year
   To improve the later scenario, we can use basis functions and perform linear regression on the basis functions instead of on feature space.

2. **What are the assumptions that are made to generate the analytical solution? Can you derive the formula for the analytical solution?**

   i. The expected value of the residual errors is zero
   ii. The residual errors are uncorrelated and share the same variance (across the input range of x):
   iii. The residual errors follow a normal distribution:

   To derive the analytical solution we need to minimize the sum of the squared errors with respect to w. Using the slides 31 and 32 of the second part of the week 2 lecture:

$$
\begin{aligned}
\arg\min_w \|\varepsilon\|^2 = \arg\min_w \|y - Xw\|^2 \\
&= (y - Xw)^T(y - Xw) \\
&= y^Ty - (Xw)^Ty - y^TXw + (Xw)^TXw \\
&= -(Xw)^Ty - y^TXw + (Xw)^TXw & (AB)^T = B^TA^T \\
&= -w^TX^Ty - y^TXw + w^TX^TXw \\
&= -2y^TXw - w^TX^TXw
\end{aligned}
$$

Using partial derivatives:

$$\frac{\delta}{\delta w} w^TAw = w^T(A + A^T)$$

$$
\begin{aligned}
0 &= -2y^TX - w^T(X^TX + (X^TX)^T) \\
&= -2y^TX + w^T(X^TX + (X^TX)^T) & \mathit{symmetric} \\
&= -2y^TX + 2w^TX^TX \\
&\Leftrightarrow y^TX = (w^TX^T)X \\
&\Leftrightarrow X^Ty = X^TXw \\
&\Leftrightarrow (X^TX)^{-1}X^Ty = w
\end{aligned}
$$

3. **Can you mention a situation where the squares error loss function would not perform well? What other loss functions can be used in that scenario?**

   In the presence of outliers the squares error loss does not perform well. To mitigate this problem, we can use other loss functions which are less senstitive to outliers like the absolute
   loss (L1) function

4. Logistic Regression

1. **What is the input and the output space of logistic regression? Hint: we are looking for something like X ∈ N**

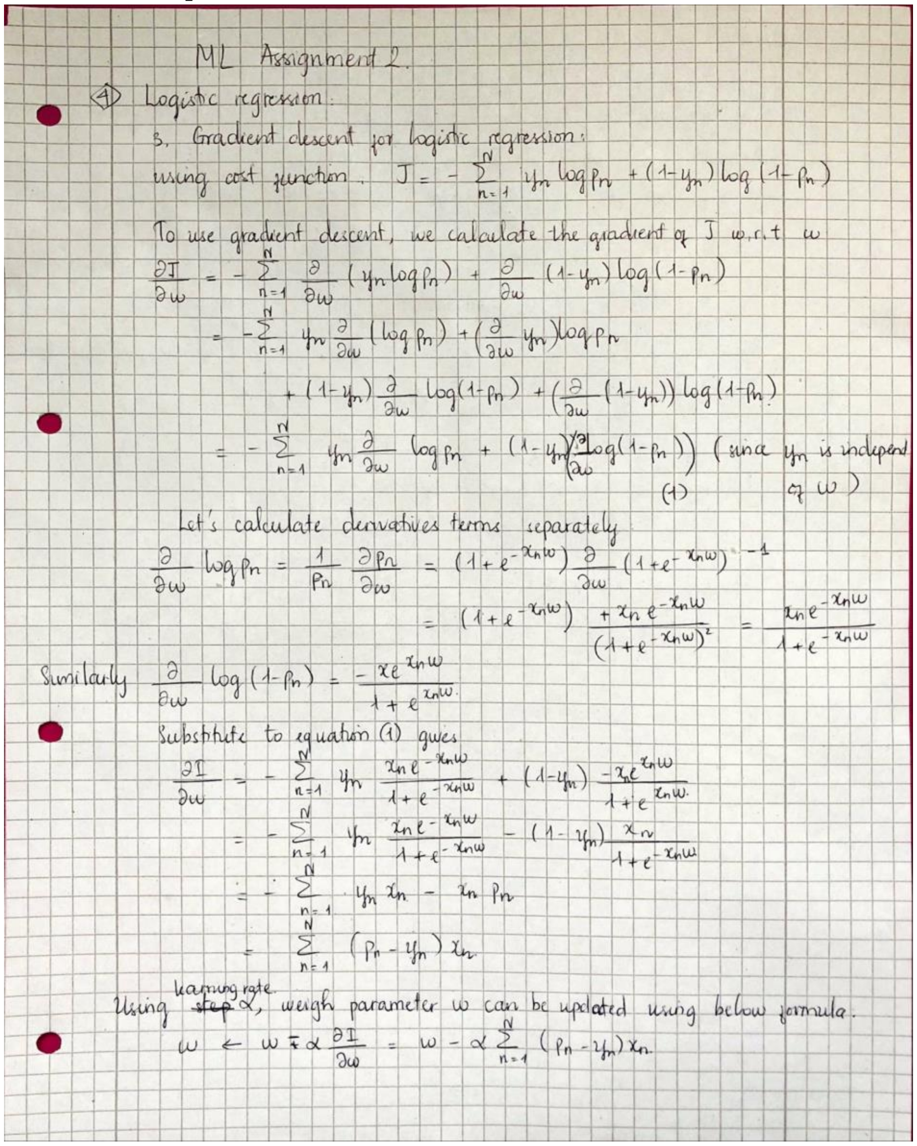
   - Input space: R^{NxD} with N is number of samples samples and D is number of features
   - Output space: $(1,0)^N$

2. **Recap gradient descent: What is it and why do we use it?**

   Gradient descent is an optimization algorithm to find the local minimum of a differentiable function. It is achieved by iteratively taking a step in the opposite direction of the gradient (first order derivative) of the function, until the gradient becomes very small

3. **Starting from negative loglikelihood (binary cross entropy loss), derive the update rule for w for gradient descent. Ignore the bias term for the moment. Why is there no closed form solution for logistic regression as there is for linear regression?**

Efficient way to estimate solution when it is impossible to derive closed-form solution or is too expensive to calculate it.



ML Assignment 2.

④ Logistic regression:

3. Gradient descent for logistic regression:

using cost function. $J = -\sum_{n=1}^{N} y_n \log p_n + (1-y_n)\log(1-p_n)$

To use gradient descent, we calculate the gradient of J w.r.t w

$$\frac{\partial J}{\partial w} = -\sum_{n=1}^{N} \frac{\partial}{\partial w}(y_n \log p_n) + \frac{\partial}{\partial w}(1-y_n)\log(1-p_n)$$

$$= -\sum_{n=1}^{N} y_n \frac{\partial}{\partial w}(\log p_n) + \left(\frac{\partial}{\partial w} y_n\right)\log p_n$$

$$+ (1-y_n)\frac{\partial}{\partial w}\log(1-p_n) + \left(\frac{\partial}{\partial w}(1-y_n)\right)\log(1-p_n)$$

$$= -\sum_{n=1}^{N} y_n \frac{\partial}{\partial w}\log p_n + (1-y_n)\left(\frac{\partial}{\partial w}\log(1-p_n)\right) \quad (1)$$ (since y_n is independ of w)

Let's calculate derivatives terms separately

$$\frac{\partial}{\partial w}\log p_n = \frac{1}{p_n}\frac{\partial p_n}{\partial w} = (1+e^{-x_n w})\frac{\partial}{\partial w}(1+e^{-x_n w})^{-1}$$

$$= (1+e^{-x_n w})\frac{+x_n e^{-x_n w}}{(1+e^{-x_n w})^2} = \frac{x_n e^{-x_n w}}{1+e^{-x_n w}}$$

Similarly $\frac{\partial}{\partial w}\log(1-p_n) = \frac{-x e^{x_n w}}{1+e^{x_n w}}$.

Substitute to equation (1) gives

$$\frac{\partial J}{\partial w} = -\sum_{n=1}^{N} y_n \frac{x_n e^{-x_n w}}{1+e^{-x_n w}} + (1-y_n)\frac{-x_n e^{x_n w}}{1+e^{x_n w}}$$

$$= -\sum_{n=1}^{N} y_n \frac{x_n e^{-x_n w}}{1+e^{-x_n w}} - (1-y_n)\frac{x_n}{1+e^{-x_n w}}$$

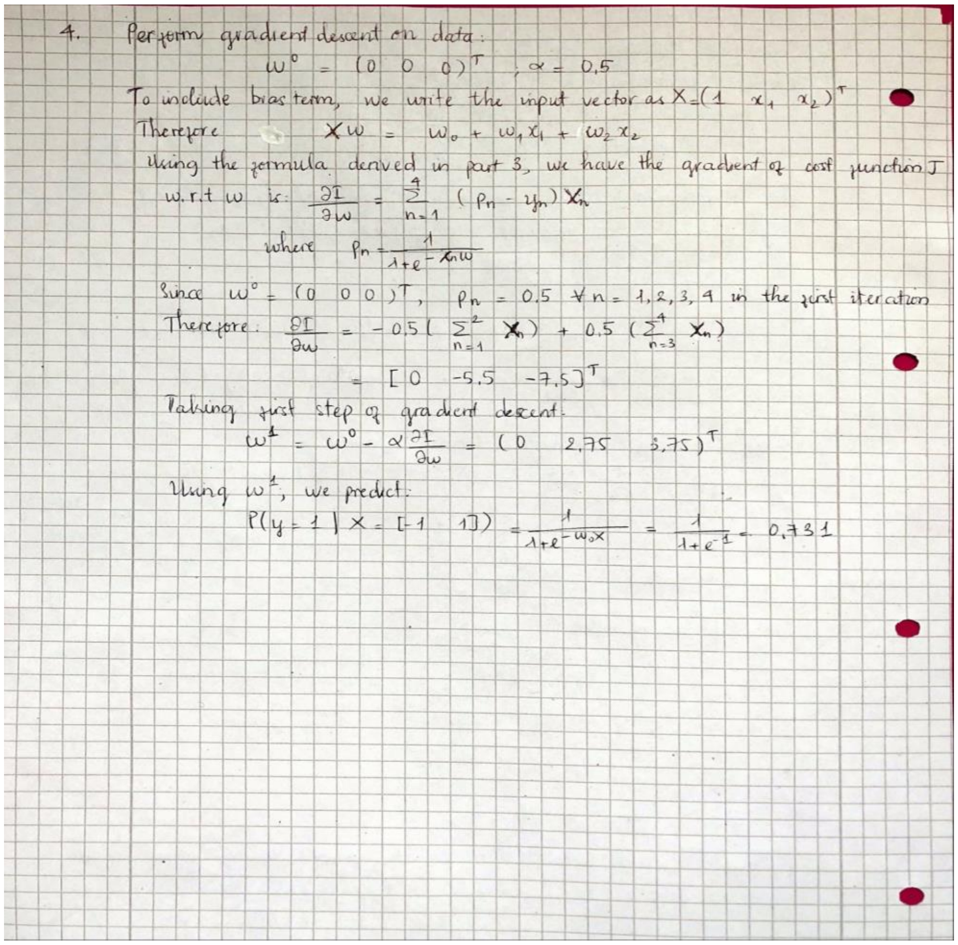
$$= -\sum_{n=1}^{N} y_n x_n - x_n p_n$$

$$= \sum_{n=1}^{N} (p_n - y_n)x_n$$

Using ~~step~~ learning rate α, weigh parameter w can be updated using below formula.

$$w \leftarrow w \mp \alpha \frac{\partial J}{\partial w} = w - \alpha \sum_{n=1}^{N}(p_n - y_n)x_n$$

4. Perform one gradient descent update step for using the initial parameters



4. Perform gradient descent on data:

$$w^0 = (0 \quad 0 \quad 0)^T \quad ; \alpha = 0{,}5$$

To include bias term, we write the input vector as $X = (1 \quad x_1 \quad x_2)^T$

Therefore $Xw = w_0 + w_1 x_1 + w_2 x_2$

Using the formula derived in part 3, we have the gradient of cost function J w.r.t w is:

$$\frac{\partial J}{\partial w} = \sum_{n=1}^{4} (p_n - y_n) X_n$$

where $p_n = \frac{1}{1+e^{-X_n w}}$

Since $w^0 = (0 \quad 0 \quad 0)^T$, $p_n = 0.5 \ \forall n = 1, 2, 3, 4$ in the first iteration

Therefore:

$$\frac{\partial J}{\partial w} = -0.5 \left(\sum_{n=1}^{2} X_n \right) + 0.5 \left(\sum_{n=3}^{4} X_n \right)$$

$$= [0 \quad -5.5 \quad -7.5]^T$$

Taking first step of gradient descent:

$$w^1 = w^0 - \alpha \frac{\partial J}{\partial w} = (0 \quad 2{,}75 \quad 3{,}75)^T$$

Using w^1, we predict:

$$P(y = 1 \mid X = [-1 \quad 1]) = \frac{1}{1+e^{-w_0 X}} = \frac{1}{1+e^{-1}} = 0{,}731$$

5. **Linear Separability**

1. **Why is this problem not linearly separable using this proposed model?**

   Because this is XOR problem. A linear model applied directly to the data can't implement the XOR function. After applying this proposal model, we obtain w=0 and b=0.5 and the outputs 0.5 everywhere.

2. **How would you modify the problem to do it linearly separable?**

   One way to solve this problem is to use a model that learns a different feature space in which a linear model is able to represent the solution. A very simple feedforward network with one hidden layer can solve it as function:

$$f(x, W, c, w, b) = w^T max\{0, W^T x + c\} + b$$

3. **Specify one set of weights that classify the input correctly.**

   We can specify a solution to this XOR problem. Let

$$W = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}, \ c = \begin{bmatrix} 0 \\ -1 \end{bmatrix}, w = \begin{bmatrix} 1 \\ -2 \end{bmatrix}, \ b = 0$$