UNIVERSITY OF AMSTERDAM

University of Amsterdam

Homework Assignment 2

Machine Learning I

2024

Pedro M. P. Curvo
15713725

Contents

1 Naive Bayes Modeling of Climate

a)

Our dataset consists of N samples, each of which composed by $\boldsymbol{x} \in \mathbb{R}^D$ features and a target variable $t \in \mathbb{R}^K$, where K is the set of possible classes. t is a one-hot encoded vector, where the k-th element is 1 if the sample belongs to class k and 0 otherwise. Thus, we can write the likelihood of the data, without assuming Naive Bayes, as:

$$
\begin{aligned}
p(\boldsymbol{T}, \boldsymbol{X}|\boldsymbol{M}, \boldsymbol{B}) &= \prod_{n=1}^{N} p(\boldsymbol{t}_n, \boldsymbol{x}_n|\boldsymbol{M}, \boldsymbol{B}) \quad \text{assuming i.i.d samples} \\
&= \prod_{n=1}^{N} p(\boldsymbol{x_n}|\boldsymbol{t_n}, \boldsymbol{M}, \boldsymbol{B}) p(\boldsymbol{t_n}|\boldsymbol{M}, \boldsymbol{B}) \quad \text{, making use of the product rule} \\
&= \prod_{n=1}^{N} \prod_{k=1}^{K} [p(\boldsymbol{x_n}|t_{nk} = 1, \boldsymbol{M}, \boldsymbol{B}) p(t_{nk} = 1|\boldsymbol{M}, \boldsymbol{B})]^{t_{nk}} \\
&= \prod_{n=1}^{N} \prod_{k=1}^{K} [p(\boldsymbol{x_n}|C_k, \boldsymbol{M}, \boldsymbol{B}) p(C_k|\boldsymbol{M}, \boldsymbol{B})]^{t_{nk}} \quad \text{, since} \quad p(C_k|\boldsymbol{M}, \boldsymbol{B}) = p(t_{nk} = 1|\boldsymbol{M}, \boldsymbol{B}) \\
&= \prod_{n=1}^{N} \prod_{k=1}^{K} [\pi_k p(\boldsymbol{x_n}|C_k, \boldsymbol{M}, \boldsymbol{B})]^{t_{nk}} \quad \text{, since} \quad p(C_k|\boldsymbol{M}, \boldsymbol{B}) = \pi_k
\end{aligned}
$$

Now, assuming the Naive Bayes assumption, we have that the features are conditionally independent given the class, i.e. $p(\boldsymbol{x_n}|C_k, \boldsymbol{M}, \boldsymbol{B}) = \prod_{d=1}^{D} p(x_{nd}|C_k, \boldsymbol{M}, \boldsymbol{B})$, which allows us to write the likelihood as:

$$
p(\boldsymbol{T}, \boldsymbol{X}|\boldsymbol{M}, \boldsymbol{B}) = \prod_{n=1}^{N} \prod_{k=1}^{K} [\pi_k \prod_{d=1}^{D} p(x_{nd}|C_k, \boldsymbol{M}, \boldsymbol{B})]^{t_{nk}}
$$

b)

Assuming that the features are modelled as Gaussian Distributions, i.e., a Multivariate Gaussian in the case of dependent features and Univariate Gaussians in the case of independent features, then we can assume the following for each one of the assumptions:

Bayes Classifier:

Since the features in $\boldsymbol{x}$ are not independent, then their relation could be explained by a multivariate Gaussian distribution. This multivariate gaussian distribution, for each class, would have a mean $\boldsymbol{\mu}_k$ and a covariance matrix $\boldsymbol{\Sigma}_k$, with $\boldsymbol{\mu}_k \in \mathbb{R}^D$ and $\boldsymbol{\Sigma}_k \in \mathbb{R}^{D \times D}$, however since $\boldsymbol{\Sigma}_k$ is a covariance matrix, it is symmetric. Thus, the total number of parameters per class would be $D + \frac{D(D+1)}{2}$, where the first term corresponds to the parameter of the mean and the second term to the ones of the covariance matrix, which are the elements of the upper triangular part of the matrix and the diagonal.

Thus, for the entire number of classes, the total number of parameters would be $K \times (D + \frac{D(D+1)}{2})$. Adding to this, we would need to consider the prior probabilities for each class, which would add $K-1$ parameters, since the last class can be inferred from the others. Thus, the total number of parameters would be:

$$K \times (D + \frac{D(D+1)}{2}) + K - 1 \quad \text{parameters, assuming a Bayes Classifier}$$

Naive Bayes Classifier:

Now, if we assume that the features are independent, and thus $p(\boldsymbol{x_n}|C_k, \boldsymbol{M}, \boldsymbol{B}) = \prod_{d=1}^{D} p(x_{nd}|C_k, \boldsymbol{M}, \boldsymbol{B})$, with $p(x_{nd}|C_k, \boldsymbol{M}, \boldsymbol{B}) = \mathcal{N}(x_{nd}|\mu_{kd}, \sigma_{kd}^2)$, since the features are modelled as Gaussian distributions.

Then, the total number of parameters per class would be $2D$:

- D parameters that correspond to the mean for each feature.
- D parameters that correspond to the variance for each feature.

We can also think this as the following: on the previous example of the Bayes Classifier, we had a covariance matrix that had $D + \frac{D(D+1)}{2}$ parameters, since the upper triangular part of the matrix was filled. Now, since the features are independent, the covariance matrix would be a diagonal matrix, which would have only D parameters, because the diagonal of $\boldsymbol{\Sigma}_k$ would be the variance of each feature and the other elements would be zero, since they correspond to the covariance between the features, which in this case: $Cov(x_{ni}, x_{nj}) = 0$ for $i \neq j$.

Thus, for the entire number of classes, the total number of parameters would be:

$$K \times 2D$$

For the prior probabilities, we would need $K-1$ parameters, as in the previous case, since the last class can be inferred from the others.

This brings the total number of parameters to:

$$K \times 2D + K - 1 \quad \text{parameters, assuming a Naive Bayes Classifier}$$

As we can see, Naive Bayes reduces the complexity of the model from a quadratic to a linear number of parameters, which makes it a more efficient model, but with the cost of the assumption that the features are independent.

Naive

The term “naive” in the Naive Bayes classifier refers to the assumption that features are conditionally independent given the class label, meaning that the features do not influence each other. This assumption usually falls in real-world, where correlated features are common. For instance, in the climate set described above, regions could be correlated based on their location. If one region is in the southern hemisphere and another in the northern hemisphere, their seasons would be inverted, which means they have a negative correlation.

Another example, starting from the weather above, could be that given two features: $X =$ temperature and $Y =$ snow. The Naive Bayes classifier would calculate the probability of both 40°C and snow occurring by multiplying their individual probabilities. In reality, this is wrong because the joint probability would be 0, as snow does not happen at that temperature and the probability is actually 0.

Another example that can be used to illustrate this is, for example, the detection of spam. i.e., imagine that you have features that are the presence of certain words in the email, for example, "win", "money", etc. You know that when they appear together, the email is more likely to be spam, thus the features are correlated.

c)

Assuming the Naive Bayes assumption, we can write the likelihood as we showed in the first question:

$$p(\boldsymbol{T}, \boldsymbol{X}|\boldsymbol{M}, \boldsymbol{B}) = \prod_{n=1}^{N}\prod_{k=1}^{K}[\pi_k \prod_{d=1}^{D} p(x_{nd}|C_k, \boldsymbol{M}, \boldsymbol{B})]^{t_{nk}}$$

Assuming the features are modelled as Gaussian distributions, we can write the likelihood as:

$$p(\boldsymbol{T}, \boldsymbol{X}|\boldsymbol{M}, \boldsymbol{B}) = \prod_{n=1}^{N}\prod_{k=1}^{K}[\pi_k \prod_{d=1}^{D} \frac{\beta_{dk}^{\frac{1}{2}}}{\sqrt{2\pi}} \exp\left(-\frac{\beta_{dk}}{2}(x_{nd} - \mu_{dk})^2\right)]^{t_{nk}}$$

Taking the logarithm of the likelihood, we have:

$$\log p(\boldsymbol{T}, \boldsymbol{X}|\boldsymbol{M}, \boldsymbol{B}) = \sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk}\left[\log \pi_k + \sum_{d=1}^{D}\left[\frac{1}{2}\log \beta_{dk} - \frac{1}{2}\log 2\pi - \frac{\beta_{dk}}{2}(x_{nd} - \mu_{dk})^2\right]\right]$$

d)

Solving for the MLE estimator for μ_{dk}, we have:

$$\frac{\partial \log p(\boldsymbol{T}, \boldsymbol{X}|\boldsymbol{M}, \boldsymbol{B})}{\partial \mu_{ij}} = 0$$

$$\frac{\partial}{\partial \mu_{ij}} \sum_{n=1}^{N} \sum_{k=1}^{K} t_{nk} \left[\log \pi_k + \sum_{d=1}^{D} \left[\frac{1}{2} \log \beta_{dk} - \frac{1}{2} \log 2\pi - \frac{\beta_{dk}}{2} (x_{nd} - \mu_{dk})^2 \right] \right] = 0$$

Since the only terms that depend on μ_{ij} are the ones when $k = j$ and $d = i$, we have:

$$\frac{\partial}{\partial \mu_{ij}} \sum_{n=1}^{N} t_{nj} \left[\log \pi_j \frac{1}{2} \log \beta_{ij} - \frac{1}{2} \log 2\pi - \frac{\beta_{ij}}{2} (x_{nj} - \mu_{ij})^2 \right] = 0$$

$$\sum_{n=1}^{N} t_{nj} \left[\beta_{ij} (x_{nj} - \mu_{ij}) \right] = 0$$

$$\sum_{n=1}^{N} t_{nj} \beta_{ij} x_{ni} - \sum_{n=1}^{N} t_{nj} \beta_{ij} \mu_{ij} = 0$$

$$\sum_{n=1}^{N} t_{nj} \beta_{ij} x_{ni} = \sum_{n=1}^{N} t_{nj} \beta_{ij} \mu_{ij}$$

$$\sum_{n=1}^{N} t_{nj} x_{ni} = \sum_{n=1}^{N} t_{nj} \mu_{ij}$$

$$\mu_{ij} = \frac{\sum_{n=1}^{N} t_{nj} x_{ni}}{\sum_{n=1}^{N} t_{nj}}$$

Hence,

$$\mu_{dk_{MLE}} = \frac{\sum_{n=1}^{N} t_{nk} x_{nd}}{\sum_{n=1}^{N} t_{nk}}$$

Thus, assuming that t_{nk} is a binary variable that takes the value 1 when $x_n \in C_k$, we can see that the MLE estimator for μ_{dk} is the average of samples that belong to class k, i.e, it represents a weighted average of the samples that belong to class k.

e)

Specifying $p(C_1|\boldsymbol{x})$ for the general k classes of the Naive Bayes classifier, we have:

$$
\begin{aligned}
p(C_1|\boldsymbol{x}) &= \frac{p(\boldsymbol{x}|C_1)p(C_1)}{p(\boldsymbol{x})} \quad \text{, by Bayes' Theorem} \\
&= \frac{p(\boldsymbol{x}|C_1)p(C_1)}{\sum_{k=1}^{K} p(\boldsymbol{x}|C_k)p(C_k)} \quad \text{, by the law of total probability} \\
&= \frac{\pi_1 \prod_{d=1}^{D} p(x_d|C_1)}{\sum_{k=1}^{K} \pi_k \prod_{d=1}^{D} p(x_d|C_k)} \quad \text{, by the Naive Bayes assumption that the features are i.i.d.} \\
&= \frac{\pi_1 \prod_{d=1}^{D} p(x_d|C_1, \mu_{d1}, \beta_{d1})}{\sum_{k=1}^{K} \pi_k \prod_{d=1}^{D} p(x_d|C_k, \mu_{dk}, \beta_{dk})} \quad \text{, by the Gaussian assumption}
\end{aligned}
$$

f)

Firstly, we need to bare in mind that, contrary to a logistic regression, which is a discriminative model and models the posterior probability $p(C_k|\boldsymbol{x})$ without the need to explicitly model the likelihood $p(\boldsymbol{x}|C_k)$, caring only about the decision boundary, the Naive Bayes classifier models the joint probability $p(C_k, \boldsymbol{x})$. Summing this idea: the Naive Bayes classifier learns the joint probability which allows to then sample from it, while the logistic regression does not care about the distribution of the data.

Looking at our previous answer, we can see that the Naive Bayes Classifier models the joint probability $p(\boldsymbol{x}, C_1) = p(\boldsymbol{x}|C_1)p(C_1)$ by first generating a class target C_1 based on the information of the prior and then generating the features $\boldsymbol{x}$ based on the information of the likelihood $p(\boldsymbol{x}|C_1)$.

g)

Writing $p(C_1|\boldsymbol{x})$ for the normally distributed model explicitly, and considering the assumptions of the previous questions, we have:

$$
\begin{aligned}
p(C_1|\boldsymbol{x}) &= \frac{p(\boldsymbol{x}|C_1)p(C_1)}{p(\boldsymbol{x})} \quad \text{, by Bayes' Theorem} \\
&= \frac{p(\boldsymbol{x}|C_1)p(C_1)}{\sum_{k=1}^{K} p(\boldsymbol{x}|C_k)p(C_k)} \quad \text{, by the law of total probability} \\
&= \frac{\pi_1 \prod_{d=1}^{D} p(x_d|C_1)}{\sum_{k=1}^{K} \pi_k \prod_{d=1}^{D} p(x_d|C_k)} \quad \text{, by the Naive Bayes assumption that the features are i.i.d.} \\
&= \frac{\pi_1 \prod_{d=1}^{D} p(x_d|C_1, \mu_{d1}, \beta_{d1})}{\sum_{k=1}^{K} \pi_k \prod_{d=1}^{D} p(x_d|C_k, \mu_{dk}, \beta_{dk})} \quad \text{, by the Gaussian assumption} \\
&= \frac{\pi_1 \prod_{d=1}^{D} \frac{\beta_{d1}^{\frac{1}{2}}}{\sqrt{2\pi}} \exp\left(-\frac{\beta_{d1}}{2}(x_d - \mu_{d1})^2\right)}{\sum_{k=1}^{K} \pi_k \prod_{d=1}^{D} \frac{\beta_{dk}^{\frac{1}{2}}}{\sqrt{2\pi}} \exp\left(-\frac{\beta_{dk}}{2}(x_d - \mu_{dk})^2\right)}
\end{aligned}
$$

h)

A datapoint $\boldsymbol{x}$ is classified as C_1 if:

$$p(C_1|\boldsymbol{x}) > p(C_k|\boldsymbol{x}) \quad ,\forall k \neq 1$$

i)

Considering the inequality $p(C_1|\boldsymbol{x}) > p(C_k|\boldsymbol{x})$, $\forall k \neq 1$, we can write it as:

$$p(C_1|\boldsymbol{x}) > p(C_k|\boldsymbol{x})$$
$$\frac{\pi_1 \prod_{d=1}^{D} p(x_d|C_1, \mu_{d1}, \beta_{d1})}{\sum_{k=1}^{K} \pi_k \prod_{d=1}^{D} p(x_d|C_k, \mu_{dk}, \beta_{dk})} > \frac{\pi_k \prod_{d=1}^{D} p(x_d|C_k, \mu_{dk}, \beta_{dk})}{\sum_{k=1}^{K} \pi_k \prod_{d=1}^{D} p(x_d|C_k, \mu_{dk}, \beta_{dk})}$$

Since the denominator is the same for both sides and it is always positive since it is a sum of probabilities and the probabilities are positive we can write it as:

$$\pi_1 \prod_{d=1}^{D} p(x_d|C_1, \mu_{d1}, \beta_{d1}) > \pi_k \prod_{d=1}^{D} p(x_d|C_k, \mu_{dk}, \beta_{dk})$$

Which, assuming the Gaussian Distributions, we can write as:

$$\pi_1 \prod_{d=1}^{D} \frac{\beta_{d1}^{\frac{1}{2}}}{\sqrt{2\pi}} \exp\left(-\frac{\beta_{d1}}{2}(x_d - \mu_{d1})^2\right) > \pi_k \prod_{d=1}^{D} \frac{\beta_{dk}^{\frac{1}{2}}}{\sqrt{2\pi}} \exp\left(-\frac{\beta_{dk}}{2}(x_d - \mu_{dk})^2\right)$$

To have quadratic inequality in the form $ax^2 + bx + c > 0$, we can take the logarithm of both sides, since if $f(x) > g(x)$, then $\log f(x) > \log g(x) \forall f(x), g(x) > 0$:

$$\log \pi_1 + \sum_{d=1}^{D} \left[\frac{1}{2}\log \beta_{d1} - \frac{1}{2}\log 2\pi - \frac{\beta_{d1}}{2}(x_d - \mu_{d1})^2\right] > \log \pi_k + \sum_{d=1}^{D} \left[\frac{1}{2}\log \beta_{dk} - \frac{1}{2}\log 2\pi - \frac{\beta_{dk}}{2}(x_d - \mu_{dk})^2\right]$$
$$\log \pi_1 + \sum_{d=1}^{D} \left[\frac{1}{2}\log \beta_{d1} - \frac{1}{2}\log 2\pi - \frac{\beta_{d1}}{2}(x_d - \mu_{d1})^2\right] - \log \pi_k - \sum_{d=1}^{D} \left[\frac{1}{2}\log \beta_{dk} - \frac{1}{2}\log 2\pi - \frac{\beta_{dk}}{2}(x_d - \mu_{dk})^2\right] > 0$$
$$\log \frac{\pi_1}{\pi_k} + \sum_{d=1}^{D} \left[\frac{1}{2}\log \frac{\beta_{d1}}{\beta_{dk}} - \frac{\beta_{d1}}{2}(x_d - \mu_{d1})^2 + \frac{\beta_{dk}}{2}(x_d - \mu_{dk})^2\right] > 0$$
$$2\log \frac{\pi_1}{\pi_k} + \sum_{d=1}^{D} \left[\log \frac{\beta_{d1}}{\beta_{dk}} - \beta_{d1}(x_d - \mu_{d1})^2 + \beta_{dk}(x_d - \mu_{dk})^2\right] > 0$$
$$\sum_{d=1}^{D} \left[-\beta_{d1}(x_d - \mu_{d1})^2 + \beta_{dk}(x_d - \mu_{dk})^2\right] > -2\log \frac{\pi_1}{\pi_k} - \sum_{d=1}^{D} \left[\log \frac{\beta_{d1}}{\beta_{dk}}\right]$$
$$\sum_{d=1}^{D} \left[-\beta_{d1}x_d^2 + 2\beta_{d1}x_d\mu_{d1} - \beta_{d1}\mu_{d1}^2 + \beta_{dk}x_d^2 - 2\beta_{dk}x_d\mu_{dk} + \beta_{dk}\mu_{dk}^2\right] > -2\log \frac{\pi_1}{\pi_k} - \sum_{d=1}^{D} \left[\log \frac{\beta_{d1}}{\beta_{dk}}\right]$$
$$\sum_{d=1}^{D} \left[x_d^2(\beta_{dk} - \beta_{d1}) + 2x_d(\beta_{d1}\mu_{d1} - \beta_{dk}\mu_{dk}) + \beta_{d1}\mu_{d1}^2 - \beta_{dk}\mu_{dk}^2\right] > -2\log \frac{\pi_1}{\pi_k} - \sum_{d=1}^{D} \left[\log \frac{\beta_{d1}}{\beta_{dk}}\right]$$
$$\sum_{d=1}^{D} \left[x_d^2(\beta_{dk} - \beta_{d1}) + 2x_d(\beta_{d1}\mu_{d1} - \beta_{dk}\mu_{dk})\right] > -2\log \frac{\pi_1}{\pi_k} - \sum_{d=1}^{D} \left[\log \frac{\beta_{d1}}{\beta_{dk}} - \beta_{d1}\mu_{d1}^2 + \beta_{dk}\mu_{dk}^2\right]$$

Which considering the constant terms as c, we have:

$$\sum_{d=1}^{D} \left[x_d^2(\beta_{dk} - \beta_{d1}) + 2x_d(\beta_{d1}\mu_{d1} - \beta_{dk}\mu_{dk}) \right] > c$$

j)

Given this definition, I do not expect the region to be convex, since the definition assumes a mantaince of the linearity, but due to the quadratic term, this linearity falls apart. This can be shown by using the Hessian of the decision boundary, but using the definition given in the question we can demonstrate it by the following:

First, as pointed out, if a region C_1 is convex for any two points $\boldsymbol{x}_1, \boldsymbol{x}_2 \in C_1$ and any $\lambda \in [0, 1]$, then all the points that lie in the line segment between $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ are also in C_1, meaning that the point:

$$\boldsymbol{x}_\lambda = \lambda \boldsymbol{x}_1 + (1 - \lambda)\boldsymbol{x}_2 \in C_1$$

Now, let $\boldsymbol{x}_1 = (x_{11}, \cdots, x_{1D})$ and $\boldsymbol{x}_2 = (x_{21}, \cdots, x_{2D})$ be two points in the region C_1, meaning that:

$$\sum_{d=1}^{D} \left[x_{1d}^2(\beta_{dk} - \beta_{d1}) + 2x_{1d}(\beta_{d1}\mu_{d1} - \beta_{dk}\mu_{dk}) \right] > c$$

and

$$\sum_{d=1}^{D} \left[x_{2d}^2(\beta_{dk} - \beta_{d1}) + 2x_{2d}(\beta_{d1}\mu_{d1} - \beta_{dk}\mu_{dk}) \right] > c$$

Now, we need to check if $\boldsymbol{x}_\lambda = \lambda \boldsymbol{x}_1 + (1 - \lambda)\boldsymbol{x}_2$ is in C_1, i.e., if:

$$\sum_{d=1}^{D} \left[x_{\lambda d}^2(\beta_{dk} - \beta_{d1}) + 2x_{\lambda d}(\beta_{d1}\mu_{d1} - \beta_{dk}\mu_{dk}) \right] > c$$

Expanding the quadratic term, we have:

$$\begin{aligned} x_{\lambda d}^2 &= (\lambda x_{1d} + (1 - \lambda)x_{2d})^2 \\ &= \lambda^2 x_{1d}^2 + 2\lambda(1 - \lambda)x_{1d}x_{2d} + (1 - \lambda)^2 x_{2d}^2 \end{aligned}$$

Expanding the linear term, we have:

$$x_{\lambda d} = \lambda x_{1d} + (1 - \lambda)x_{2d}$$

Now, substituting the expanded terms in the inequality, we have:

$$\sum_{d=1}^{D} \left[\left(\lambda^2 x_{1d}^2 + 2\lambda(1 - \lambda)x_{1d}x_{2d} + (1 - \lambda)^2 x_{2d}^2 \right)(\beta_{dk} - \beta_{d1}) + 2\left(\lambda x_{1d} + (1 - \lambda)x_{2d}\right)(\beta_{d1}\mu_{d1} - \beta_{dk}\mu_{dk}) \right] > c$$

Now, we see that we could decompose in a linear combination of the previous inequalities for $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$, which would respect the inequality, however, the quadratic is a cross term between x_{1d} and x_{2d}, making it non-linear separable, hence, the region is not convex, i.e., $\exists \lambda \in [0, 1]$ such that $x_\lambda \notin C_1$.

Taking the Hessian for the decision is a more straightforward way to evaluate the convexity, but it will be used in the next question.

k)

Considering the inequality showed before, we should expect that since it is a quadratic inequality, the decision boundary would create a region that is not convex. If we eliminate the quadratic term, we would have a linear inequality, which would create a linear decision boundary, which is true if we consider the case where $\beta_{dk} = \beta_{dl}$ for $\forall k \neq l$. However, I'm going to show this in the next paragraphs.

The decision boundary is the set of points where $p(C_1|\boldsymbol{x}) = p(C_k|\boldsymbol{x})$, $\forall k \neq 1$.

Now, considering the inequality showed before we can say that the boundary for the region that a point x belongs to class C_1 or C_k is the set of points where the inequality becomes:

$$\sum_{d=1}^{D} \left[x_d^2(\beta_{dk} - \beta_{d1}) + 2x_d(\beta_{d1}\mu_{d1} - \beta_{dk}\mu_{dk}) \right] = c$$

Now, to evaluate the shape of the decision boundary, we can take the Hessian of the condition:

$$H = \begin{bmatrix} \frac{\partial^2}{\partial x_1^2} & \frac{\partial^2}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2}{\partial x_1 \partial x_D} \\ \frac{\partial^2}{\partial x_2 \partial x_1} & \frac{\partial^2}{\partial x_2^2} & \cdots & \frac{\partial^2}{\partial x_2 \partial x_D} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2}{\partial x_D \partial x_1} & \frac{\partial^2}{\partial x_D \partial x_2} & \cdots & \frac{\partial^2}{\partial x_D^2} \end{bmatrix}$$

$$H = \begin{bmatrix} 2(\beta_{1k} - \beta_{11}) & 0 & \cdots & 0 \\ 0 & 2(\beta_{2k} - \beta_{21}) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 2(\beta_{Dk} - \beta_{D1}) \end{bmatrix}$$

Now, if the Hessian is positive definite, then the decision boundary is convex, otherwise, it is not. Hence, we need to evaluate the eigenvalues of the Hessian matrix, which, since it is a diagonal matrix, are the elements of the diagonal.

- If the eigenvalues are all non negative, then the decision region inside the boundary is a convex region.
- If the eigenvalues are all non positive, then the decision region inside the boundary is not a convex region.
- If the eigenvalues are all zero, then the decision region is a linear region.

This lead to the conclusion, only for part of the decision regions that separate the classes, that:

- if $\beta_{dk} > \beta_{d1} \forall d$, then the decision boundary is convex, meaning one of the regions is convex and, hence, the other is non-convex.
- if $\beta_{dk} < \beta_{d1} \forall d$, then the decision boundary is non-convex, meaning one of the regions is non-convex and, hence, the other is convex.
- if $\beta_{dk} = \beta_{d1} \forall d$, then the decision boundary is linear, and both regions are convex.

Note: The boundary is a function, hence it can be convex or concave, however, both options lead to a convex and a non-convex region, because if the boundary is convex, then the region inside the boundary is convex and the region outside the boundary is non-convex, and vice-versa. If it's linear, then both regions are convex. Also, we need to bare in mind that we are only evaluating the regions

that separate the classes, and not the overall problem, if the region is convex relative to class C_2, it does not mean that it is convex relative to class C_3.

Well, now we need to consider the overall problem and see all the decision regions. For that we need to have in mind that if all the subboundaries of a class C_k are convex, then the decision region for that class is convex, and if one of the subboundaries is non-convex, then the decision region is non-convex. With this, we can say that, unless all the decision boundaries are linear, we can have at most one convex region and $K-1$ non-convex regions. because if one region is convex relative to all the other classes, it means that all the other classes would have a boundary where the region is non-convex. Meaning, that in general, and without the assumption of the hint, the problem becomes non-convex.

Now, considering the hint, we have that $\beta_{d1} = \beta_{dk}$, $\forall d, k$, then the decision boundaries are linear, and, hence we would end with a set of linar decision regions which would all be convex.

This also makes sense relating to the previous question that was meant to show the quadratic inequality that would define the decision boundary. If we consider the case where $\beta_{dk}! = \beta_{dl}$ for $k \neq l$, then the inequality would be a quadratic inequality. Whereas, if we consider the case where $\beta_{dk} = \beta_{dl}$ for $k \neq l$, then the inequality would be a linear inequality.

2 Binary Classification

a)

i)

The dataset above **cannot** be classified using a logistic regression model with linear features, since the data is **not linearly separable**, meaning that there is no line that can separate the two classes.

Assuming we have points $A(1,1)$, $B(-1,-1)$, $C(1,-1)$ and $D(-1,1)$, we can see that the points A and B belong to class *Blue* and the points C and D belong to class *Orange*. If we now assume a line that separates the two classes, this boundary could be defined as $ax+by+c=0$, where a, b and c are the parameters of the line. We would know that for the points to be on the same side of the line, then:

$$(ax_1 + by_1 + c) \cdot (ax_2 + by_2 + c) > 0$$

This because if a point belongs to a class, let's say the class above the line, putting in the equation of the line would give a positive value, and the same for the other point. For the other class, the value would be negative. Hence, we know that the product of the two values should be positive, if the points belong to the same class. Saying this, considering points A and B, we would have:

$$(a + b + c) \cdot (-a - b + c) > 0$$

Now, considering points A and D, we would have:

$$(a + b + c) \cdot (a - b - c) > 0$$

Now, considering points D and C, we would have:

$$(-a + b + c) \cdot (a - b + c) > 0$$

We can combain them to have:

$$(a + b + c)^2 \cdot (-a - b + c) \cdot (a - b - c) > 0$$

Meaning, that $(-a-b+c)$ and $(a-b-c)$ should have the same sign, as $(-a+b+c)$ and $(a-b+c)$ should have the same sign, which gives us a contradiction. Meaning that the data is not linearly separable.

ii)

The data **can** be classified using a logistic regression model with **non-linear basis functions**, depending on the basis functions that are used.

For example, consider the function $\Phi \in \mathbb{R}^2 \rightarrow \mathbb{R}^3$ defined as $\Phi(x_1, x_2) = (x_1, x_2, x_1x_2)$. Now, we know that to belong to class *Orange*, the value of x_1x_2 should be negative, and to belong to class *Blue*, the value of x_1x_2 should be positive, meaning that in this higher space the data is linearly separable by the plane $x_1x_2 = 0$, which is the third dimension of the space. We could also use $\Phi \in \mathbb{R}^2 \rightarrow \mathbb{R}^1$ defined as $\Phi(x_1, x_2) = (x_1x_2)$, which would also make the data linearly separable, or even other basis functions. Basically, the idea is to transform the data into a different space where it is linearly separable, and then use the logistic regression model to classify the data.

iii)

The data **can** be classified using a **Multilayer Perceptron with one hidden layer**. While a single Perceptron is not able to classify the data, since the data is not linearly separable as shown before, a Multilayer Perceptron with one hidden layer can classify the data, since it can learn non-linear decision boundaries. For example, let's look at the architecture of a MLP with one hidden layer with 2 neurons. Neuron 1 could activate for the inputs that belong to class *Orange* and neuron 2 could activate for the inputs that belong to class *Blue*. i.e., two neurons would create two linear decision boundaries, which would allow the MLP to classify the data.

b)

The data *cannot* be classified using a *Naive Bayes Classifier* without changing the Basis. This happens because the Naive Bayes Classifier assumes that the features are independent given the class. However, in this case, if we know the class and know the value of x_1, we can infer the sign of x_2, and vice-versa, i.e., the features are not independent. For example, if we know that x belongs to class *Orange*, and we know the sign of x_1, we can infer the sign of x_2, which breaks the assumption of NB.

Given the subset of points $A(1,1)$, $B(-1,-1)$, $C(1,-1)$ and $D(-1,1)$, we can see that the points A and B belong to class *Blue* and the points C and D belong to class *Orange*. Now, assuming only this subset of points and knowing that Naive Bayes assumes that:

$$p(x_1, x_2|C_k) = p(x_1|C_k)p(x_2|C_k)$$

Then, for class orange and assuming a frequentist approach on the probabilities, we have that:

$$p(x_1 = 1|C_{\text{Orange}}) = \frac{1}{2} \quad \text{, since we have 1 sample with } x_1 = -1 \text{ and 1 sample with } x_1 = 1$$

$$p(x_2 = 1|C_{\text{Orange}}) = \frac{1}{2} \quad \text{, since we have 1 sample with } x_2 = -1 \text{ and 1 sample with } x_2 = 1$$

$$p(x_1 = 1, x_2 = 1|C_{\text{Orange}}) = p(x_1 = 1|C_{\text{Orange}})p(x_2 = 1|C_{\text{Orange}}) = \frac{1}{4} \quad \text{, assuming the Naives Bayes assumption}$$

However, the probability of $p(x_1 = 1, x_2 = 1|C_{\text{Orange}})$ is 0, since we do not have any sample that belongs to class *Orange*.

This shows that the assumption of the Naive Bayes Classifier falls, hence we cannot use it to classify the data.

However, if we do previously a change of basis, for example to the basis $\Phi(x_1, x_2) = (r, \tan\theta)$, where $r = \sqrt{x_1^2 + x_2^2}$ and $\theta = \arctan\left(\frac{x_2}{x_1}\right)$, then the data becomes linearly separable and the features become independent, e.g., if you know the radius of the point, you can infer nothing about the angle of the point and vice-versa. Considering the subset of points $A(1,1)$, $B(-1,-1)$, $C(1,-1)$ and $D(-1,1)$. Transforming to the new basis, we have that:

$A(1,1) \to A(\sqrt{2},1)$, $B(-1,-1) \to B(\sqrt{2},1)$, $C(1,-1) \to C(\sqrt{2},-1)$ and $D(-1,-1) \to D(\sqrt{2},-1)$.

Now, $p(A|C_{\text{Orange}}) = p(x_1 = \sqrt{2}|C_{\text{Orange}})p(x_2 = 1|C_{\text{Orange}}) = 1 \cdot 1 = p(x_1 = \sqrt{2}, x_2 = 1|C_{\text{Orange}})$
And we will get the same for all the other points.

c)

The dataset consists of a inner circle of points that belong to class *Orange* and an outer annulus of points that belong to class *Purple*. If we perform a change of basis to polar coordinates, keeping the dimension of the problem, we can see that the data is linearly separable, since the radius of the points that belong to class *Orange* is smaller than the radius of the points that belong to class *Purple* and the angle of the points does not matter. Hence, we can say that knowing the angle of the point gives no information about the radius and vice-versa, which makes the data linearly separable and independent in polar coordinates. This is the same reasoning as before, if we change the basis to a basis where the features are independent given the class, then the data becomes linearly separable in an independent way and the Naive Bayes Classifier can be applied.

Putting in on a mathematical form:

$$r = \sqrt{x^2 + y^2}$$
$$\theta = \arctan\left(\frac{y}{x}\right)$$

For class *Orange*, we have that:

$$r \in [0, r_1]$$
$$\theta \in [0, 2\pi]$$

For class *Purple*, we have that:

$$r \in [r_1, r_2]$$
$$\theta \in [0, 2\pi]$$

Assuming we have uniform distributions for the radius and the angle, we now have that:

$$p(r|C_{\text{Orange}}) = \begin{cases} \frac{1}{r_1} & \text{if } r \in [0, r_1] \\ 0 & \text{otherwise} \end{cases}$$
$$p(\theta|C_{\text{Orange}}) = \frac{1}{2\pi}$$
$$p(r|C_{\text{Purple}}) = \begin{cases} \frac{1}{r_2 - r_1} & \text{if } r \in [r_1, r_2] \\ 0 & \text{otherwise} \end{cases}$$
$$p(\theta|C_{\text{Purple}}) = \frac{1}{2\pi}$$

Now, in Naive Bayes we assume that the features are independent given the class, hence we have that:

$$p(r, \theta | C_{\text{Orange}}) = p(r | C_{\text{Orange}}) p(\theta | C_{\text{Orange}})$$
$$p(r, \theta | C_{\text{Purple}}) = p(r | C_{\text{Purple}}) p(\theta | C_{\text{Purple}})$$

Now, applying the Bayes rule, we have that:

$$p(C_{\text{Orange}} | r, \theta) = \frac{p(r, \theta | C_{\text{Orange}}) p(C_{\text{Orange}})}{p(r, \theta)}$$
$$p(C_{\text{Purple}} | r, \theta) = \frac{p(r, \theta | C_{\text{Purple}}) p(C_{\text{Purple}})}{p(r, \theta)}$$

Assuming Naive Bayes, we have that:

$$p(C_{\text{Orange}} | r, \theta) = \frac{p(r | C_{\text{Orange}}) p(\theta | C_{\text{Orange}}) p(C_{\text{Orange}})}{p(r, \theta)}$$
$$p(C_{\text{Purple}} | r, \theta) = \frac{p(r | C_{\text{Purple}}) p(\theta | C_{\text{Purple}}) p(C_{\text{Purple}})}{p(r, \theta)}$$

Now, substituting with the distributions, we have that:

$$p(C_{\text{Orange}} | r, \theta) = \begin{cases} \frac{\frac{1}{r_1} \frac{1}{2\pi} p(C_1)}{p(r, \theta)} & \text{if } r \in [0, r_1] \\ 0 & \text{otherwise} \end{cases}$$
$$p(C_{\text{Purple}} | r, \theta) = \begin{cases} \frac{\frac{1}{r_2 - r_1} \frac{1}{2\pi} p(C_2)}{p(r, \theta)} & \text{if } r \in [r_1, r_2] \\ 0 & \text{otherwise} \end{cases}$$

Which simplifies to the fact that we only need to know the radius of the point to classify it and the angle does not give any information about the class of the point.

d)

In **Logistic Regression with Nonlinear Basis Functions**, $\phi(x)$ is a fixed and predetermined function that is applied to the input features, i.e., this transformation does not change during the training process and the weights are learned in the transformed space, meaning it only learns the weights associated with the transformed features.

In **Multilayer Perceptron with One Hidden Layer**, the ϕ transformation is learned during the training process, along its hidden layers. The MLP adjusts the weights and biases within each layer allowing it to learn complex transformations that are the cumulative effect of the transformations learned in each layer. This allows the MLP to learn more complex decision boundaries than the Logistic Regression, since it is not bound to the fixed transformation of the features, that, in some cases, might even be unknown or hard to determine.

3 Regularized Logistic Regression

a)

Considering a logistic regression for K classes with N training vectores $\boldsymbol{x}_n$ mapped by a function $\phi(x_n)$ to M-dimensional space, given that $p(C_k|\phi(x), \boldsymbol{w_1}, \cdots, \boldsymbol{w_K}) = y_k(\phi) = \frac{\exp(w_k^T \phi)}{\sum_{j=1}^{K} \exp(w_j^T \phi)}$ and given an assumption of a Gaussian prior on the vectors $\boldsymbol{w_1}, \cdots, \boldsymbol{w_K}$:

$$p(\boldsymbol{w_1}, \cdots, \boldsymbol{w_K}|\alpha) = \mathcal{N}(w_k|0, \alpha^{-1}I)$$

We can then write the likelihood $p(T|\Phi, \boldsymbol{w_1}, \cdots, \boldsymbol{w_K})$ as:

$$\begin{aligned}
p(T|\Phi, \boldsymbol{w_1}, \cdots, \boldsymbol{w_K}) &= \prod_{n=1}^{N} p(t_n|\phi_n, \boldsymbol{w_1}, \cdots, \boldsymbol{w_K}) \quad \text{(since the data is i.i.d.)} \\
&= \prod_{n=1}^{N} \prod_{k=1}^{K} p(t_{nk} = 1|\phi_n, \boldsymbol{w_1}, \cdots, \boldsymbol{w_K})^{t_{nk}} \\
&= \prod_{n=1}^{N} \prod_{k=1}^{K} p(C_k|\phi_n, \boldsymbol{w_1}, \cdots, \boldsymbol{w_K})^{t_{nk}} \quad \text{(since } t_{nk} = 1 \text{ if } x_n \text{ belongs to class } k) \\
&= \prod_{n=1}^{N} \prod_{k=1}^{K} y_k(\phi_n)^{t_{nk}} \\
&= \prod_{n=1}^{N} \prod_{k=1}^{K} \left(\frac{\exp(\boldsymbol{w_k}^T \phi_n)}{\sum_{j=1}^{K} \exp(\boldsymbol{w_j}^T \phi_n)} \right)^{t_{nk}}
\end{aligned}$$

Writing down the log likelihood, we have:

$$\begin{aligned}
\log p(T|\Phi, \boldsymbol{w_1}, \cdots, \boldsymbol{w_K}) &= \sum_{n=1}^{N} \sum_{k=1}^{K} t_{nk} \log y_k(\phi_n) \\
&= \sum_{n=1}^{N} \sum_{k=1}^{K} t_{nk} \log \left(\frac{\exp(\boldsymbol{w_k}^T \phi_n)}{\sum_{j=1}^{K} \exp(\boldsymbol{w_j}^T \phi_n)} \right) \\
&= \sum_{n=1}^{N} \sum_{k=1}^{K} t_{nk} \left(\boldsymbol{w_k}^T \phi_n - \log \sum_{j=1}^{K} \exp(\boldsymbol{w_j}^T \phi_n) \right)
\end{aligned}$$

b)

The prior in the question is given as:

$$\begin{aligned} p(\boldsymbol{w_1}, \cdots, \boldsymbol{w_K}|\alpha) &= \prod_{k=1}^{K} \mathcal{N}(\boldsymbol{w_k}|0, \alpha^{-1}\boldsymbol{I}) \\ &= \prod_{k=1}^{K} \left(\frac{\alpha}{2\pi}\right)^{\frac{M}{2}} \exp\left(-\frac{\alpha}{2}\boldsymbol{w_k^T}\boldsymbol{w_k}\right) \\ &= \left(\frac{\alpha}{2\pi}\right)^{\frac{M}{2}K} \exp\left(-\frac{\alpha}{2}\sum_{k=1}^{K}\boldsymbol{w_k^T}\boldsymbol{w_k}\right) \end{aligned}$$

Writing down the log prior, we have:

$$\begin{aligned} \log p(\boldsymbol{w_1}, \cdots, \boldsymbol{w_K}|\alpha) &= \log\left(\left(\frac{\alpha}{2\pi}\right)^{\frac{M}{2}K} \exp\left(-\frac{\alpha}{2}\sum_{k=1}^{K}\boldsymbol{w_k^T}\boldsymbol{w_k}\right)\right) \\ &= \frac{M}{2}K\log\frac{\alpha}{2\pi} - \frac{\alpha}{2}\sum_{k=1}^{K}\boldsymbol{w_k^T}\boldsymbol{w_k} \\ &= \frac{M}{2}K\log\frac{\alpha}{2\pi} - \frac{\alpha}{2}\sum_{k=1}^{K}\sum_{m=0}^{M-1} w_{km}^2 \end{aligned}$$

Computing the logarithm helps us in two ways when we are doing the computations. Firstly, it avoids numerical underflows specially when dealing with small numbers that can arise due to small probabilities being multiplied, and secondly, it helps in the computations since the logarithm of a product is the sum of the logarithms of the terms, which makes the computations easier and, sometimes, reduce the problem to a multiplication of matrices.

c)

The posterior distribution is given by:

$$
\begin{aligned}
p(\boldsymbol{w_1}, \cdots, \boldsymbol{w_K}|T, \Phi, \alpha) &\propto p(T|\Phi, \boldsymbol{w_1}, \cdots, \boldsymbol{w_K}) p(\boldsymbol{w_1}, \cdots, \boldsymbol{w_K}|\alpha) \quad \text{(by Bayes' theorem)} \\
&= \prod_{n=1}^{N} \prod_{k=1}^{K} y_k(\phi_n)^{t_{nk}} \prod_{k=1}^{K} \left(\frac{\alpha}{2\pi}\right)^{\frac{M}{2}} \exp\left(-\frac{\alpha}{2} \boldsymbol{w_k}^T \boldsymbol{w_k}\right) \\
&= \prod_{n=1}^{N} \prod_{k=1}^{K} \left(\frac{\exp(\boldsymbol{w_k}^T \phi_n)}{\sum_{j=1}^{K} \exp(\boldsymbol{w_j}^T \phi_n)}\right)^{t_{nk}} \prod_{k=1}^{K} \left(\frac{\alpha}{2\pi}\right)^{\frac{M}{2}} \exp\left(-\frac{\alpha}{2} \boldsymbol{w_k}^T \boldsymbol{w_k}\right) \\
&= \prod_{n=1}^{N} \prod_{k=1}^{K} \left(\frac{\exp(\boldsymbol{w_k}^T \phi_n)}{\sum_{j=1}^{K} \exp(\boldsymbol{w_j}^T \phi_n)}\right)^{t_{nk}} \left(\frac{\alpha}{2\pi}\right)^{\frac{M}{2}K} \exp\left(-\frac{\alpha}{2} \sum_{k=1}^{K} \boldsymbol{w_k}^T \boldsymbol{w_k}\right) \\
&= \prod_{n=1}^{N} \prod_{k=1}^{K} \left(\frac{\exp(\boldsymbol{w_k}^T \phi_n)}{\sum_{j=1}^{K} \exp(\boldsymbol{w_j}^T \phi_n)}\right)^{t_{nk}} \left(\frac{\alpha}{2\pi}\right)^{\frac{M}{2}K} \exp\left(-\frac{\alpha}{2} \sum_{k=1}^{K} \sum_{m=0}^{M-1} w_{km}^2\right)
\end{aligned}
$$

Note: Here, we used the fact that the posterior distribution is proportional to the product of the likelihood and the prior, since the denominator of the Bayes' theorem is a constant that does not depend on the weights. Besides that, it is hard to compute due to their integral nature and in the end it would be just a normalization constant that would not affect the weights. The result would be that:

$$
p(\boldsymbol{w_1}, \cdots, \boldsymbol{w_K}|T, \Phi, \alpha) = \frac{p(T|\Phi, \boldsymbol{w_1}, \cdots, \boldsymbol{w_K}) p(\boldsymbol{w_1}, \cdots, \boldsymbol{w_K}|\alpha)}{p(T|\Phi, \alpha)}
$$

With

$$
p(T|\Phi, \alpha) = \int p(T|\Phi, \boldsymbol{w_1}, \cdots, \boldsymbol{w_K}) p(\boldsymbol{w_1}, \cdots, \boldsymbol{w_K}|\alpha) \boldsymbol{dw_1}, \cdots, \boldsymbol{dw_K}
$$

d)

To find the maximum of the posterior distribution, we would need to find:

$$argmax_{\boldsymbol{w_1},\cdots,\boldsymbol{w_K}} p(\boldsymbol{w_1},\cdots,\boldsymbol{w_K}|T,\Phi,\alpha)$$

Which is equal to find the maximum of the log posterior distribution:

$$\begin{aligned} &argmax_{\boldsymbol{w_1},\cdots,\boldsymbol{w_K}} \log p(\boldsymbol{w_1},\cdots,\boldsymbol{w_K}|T,\Phi,\alpha) \\ &argmax_{\boldsymbol{w_1},\cdots,\boldsymbol{w_K}} \log p(T|\Phi,\boldsymbol{w_1},\cdots,\boldsymbol{w_K}) + \log p(\boldsymbol{w_1},\cdots,\boldsymbol{w_K}|\alpha) \end{aligned}$$

We can ignore the log of the evidence term, since it does not depend on the weights. This then give us:

$$\begin{aligned} &argmax_{\boldsymbol{w_1},\cdots,\boldsymbol{w_K}} \log p(\boldsymbol{w_1},\cdots,\boldsymbol{w_K}|T,\Phi,\alpha) \\ &= argmax_{\boldsymbol{w_1},\cdots,\boldsymbol{w_K}} \left(\sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk} \log y_k(\phi_n) + \frac{M}{2} K \log \frac{\alpha}{2\pi} - \frac{\alpha}{2} \sum_{k=1}^{K}\sum_{m=0}^{M-1} w_{km}^2 \right) \end{aligned}$$

Now, we can drop the constant terms since they do not depend on the weights:

$$\begin{aligned} &argmax_{\boldsymbol{w_1},\cdots,\boldsymbol{w_K}} \log p(\boldsymbol{w_1},\cdots,\boldsymbol{w_K}|T,\Phi,\alpha) \\ &= argmax_{\boldsymbol{w_1},\cdots,\boldsymbol{w_K}} \left(\sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk} \log y_k(\phi_n) - \frac{\alpha}{2} \sum_{k=1}^{K}\sum_{m=0}^{M-1} w_{km}^2 \right) \end{aligned}$$

Taking the argmax of the log posterior distribution is equivalent to taking the argmin of the negative log posterior distribution, hence we have:

$$\begin{aligned} &argmin_{\boldsymbol{w_1},\cdots,\boldsymbol{w_K}} - \log p(\boldsymbol{w_1},\cdots,\boldsymbol{w_K}|T,\Phi,\alpha) \\ &= argmin_{\boldsymbol{w_1},\cdots,\boldsymbol{w_K}} \left(-\sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk} \log y_k(\phi_n) + \frac{\alpha}{2} \sum_{k=1}^{K}\sum_{m=0}^{M-1} w_{km}^2 \right) \end{aligned}$$

Which reduces the problem of minimizing the function:

$$E(\boldsymbol{w_1},\cdots,\boldsymbol{w_K}) = -\sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk} \log y_k(\phi_n) + \frac{\alpha}{2} \sum_{k=1}^{K}\sum_{m=0}^{M-1} |w_{km}|^2$$

with respect to $\boldsymbol{w_1},\cdots,\boldsymbol{w_K}$ as we wanted to show.

e)

If I'm certain that my weights should lie around zero, then I would choose a high value for α. This because if I'm certain about my weights and if they are modelled as a Gaussian distribution, then I want the most part of the distribution to be around the mean (zero), which means a smaller variance. Since the variance is inversely proportional to α, then I would choose a high value for α. This would make the weights to be close to zero, because since we are trying to minimize the function in the previous question, the term $\frac{\alpha}{2}\sum_{k=1}^{K}\sum_{m=0}^{M-1} w_{km}^2$ would also be minimized. The term is always positive, hence if α is high, then the weights would need to be close to zero to minimize the function and balance the increase in the term $\frac{\alpha}{2}\sum_{k=1}^{K}\sum_{m=0}^{M-1} w_{km}^2$ done by the high value of α.

f)

If I'm very uncertain about my weights, then the distribution for the weights should have a high variance, resembling a uniform distribution. This would mean that α should be small, since the variance is inversely proportional to α. The more uncertain I'm, the more α is small, so as $\alpha \to 0$, the equation at (d):

$$\lim_{\alpha \to 0} -\log p(\boldsymbol{w_1}, \cdots, \boldsymbol{w_K} | T, \Phi, \alpha) = \sum_{n=1}^{N} \sum_{k=1}^{K} t_{nk} \log \left(\frac{\exp(\boldsymbol{w_k}^T \phi_n)}{\sum_{j=1}^{K} \exp(\boldsymbol{w_j}^T \phi_n)} \right)$$

Which is actually the MLE solution.

g)

Taking the gradient of the log-likelihood with respect to $\boldsymbol{w_j}$:

$$
\begin{aligned}
\nabla_{\boldsymbol{w_j}} \log p(T, \Phi, \boldsymbol{w_1}, \cdots, \boldsymbol{w_K}) &= \nabla_{\boldsymbol{w_j}} \left(\sum_{n=1}^{N} \sum_{k=1}^{K} t_{nk} \log y_k(\boldsymbol{\phi_n}) \right) \\
&= \sum_{n=1}^{N} \sum_{k=1}^{K} t_{nk} \nabla_{\boldsymbol{w_j}} \log y_k(\boldsymbol{w_k^T \phi_n}) \\
&= \sum_{n=1}^{N} \sum_{k=1}^{K} t_{nk} \frac{1}{y_k(\boldsymbol{w_k^T \phi_n})} \nabla_{\boldsymbol{w_j}} y_k(\boldsymbol{w_k^T \phi_n}) \quad \text{, since y is the softmax function} \\
&= \sum_{n=1}^{N} \sum_{k=1}^{K} t_{nk} \frac{1}{y_k(\boldsymbol{w_k^T \phi_n})} y_k(\boldsymbol{w_k^T \phi_n})(\mathbb{1}_{kj} - y_k(\boldsymbol{w_k^T \phi_n})) \nabla_{\boldsymbol{w_j}} \boldsymbol{w_k^T \phi_n^T} \\
&= \sum_{n=1}^{N} \sum_{k=1}^{K} t_{nk} (\mathbb{1}_{kj} - y_j(\boldsymbol{w_j^T \phi_n})) \boldsymbol{\phi_n^T} \\
&= \sum_{n=1}^{N} \sum_{k=1}^{K} t_{nk} \mathbb{1}_{kj} \boldsymbol{\phi_n^T} - \sum_{n=1}^{N} \sum_{k=1}^{K} t_{nk} y_j(\boldsymbol{w_j^T \phi_n}) \boldsymbol{\phi_n^T} \\
&= \sum_{n=1}^{N} t_{nj} \boldsymbol{\phi_n^T} - \sum_{n=1}^{N} y_j(\boldsymbol{w_j^T \phi_n}) \boldsymbol{\phi_n^T} \sum_{k=1}^{K} t_{nk} \\
&= \sum_{n=1}^{N} t_{nj} \boldsymbol{\phi_n^T} - \sum_{n=1}^{N} y_j(\boldsymbol{w_j^T \phi_n}) \boldsymbol{\phi_n^T} \quad \text{, since } \sum_{k=1}^{K} t_{nk} = 1 \\
&= \sum_{n=1}^{N} (t_{nj} - y_j(\boldsymbol{w_j^T \phi_n})) \boldsymbol{\phi_n^T}
\end{aligned}
$$

h)

Now taking the gradient of the log-prior:

$$\begin{aligned}
\nabla_{w_j} \log p(\boldsymbol{w_1}, \cdots, \boldsymbol{w_K}|\alpha) &= \nabla_{w_j} \left(\frac{M}{2} K \log \frac{\alpha}{2\pi} - \frac{\alpha}{2} \sum_{k=1}^{K} \sum_{m=0}^{M-1} w_{km}^2 \right) \\
&= \nabla_{w_j} \left(-\frac{\alpha}{2} \sum_{m=0}^{M-1} w_{jm}^2 \right) \\
&= \nabla_{w_j} \left(-\frac{\alpha}{2} \boldsymbol{w_j}^T \boldsymbol{w_j} \right) \\
&= -\alpha \boldsymbol{w_j}^T
\end{aligned}$$

This gives a row vector, as expected since we are taking the gradient of a scalar that comes from the distribution with respect to a vector.

Now, taking the gradient of the log-posterior is just the sum of the gradients of the log-likelihood and the log-prior, since the gradient is a linear operator:

$$\begin{aligned}
\nabla_{w_j} \log p(\boldsymbol{w_1}, \cdots, \boldsymbol{w_K}|T, \Phi, \alpha) &= \nabla_{w_j} \log p(T, \Phi, \boldsymbol{w_1}, \cdots, \boldsymbol{w_K}) + \nabla_{w_j} \log p(\boldsymbol{w_1}, \cdots, \boldsymbol{w_K}|\alpha) \\
&= \sum_{n=1}^{N} (t_{nj} - y_j(\boldsymbol{w_j}^T \boldsymbol{\phi_n})) \boldsymbol{\phi_n}^T - \alpha \boldsymbol{w_j}^T
\end{aligned}$$