(1)

(a) $p(T, X \mid M, B) \overset{iid}{=} \prod_{n=1}^{N} p(\vec{t_n}, \vec{x_n} \mid M, B) =$

$\overset{\text{Prod Rule}}{=} \prod_{n=1}^{N} p(\vec{x_n} \mid \vec{t_n}, M, B) \cdot p(\vec{t_n} \mid M, B) =$

$= \prod_{n=1}^{N} \prod_{k=1}^{K} \left[p(\vec{x_n} \mid t_{nk}=1, \vec{\mu_k}, \vec{P_k}) \cdot p(t_{nk}=1 \mid \vec{\mu_k}, \vec{P_k}) \right]^{t_{nk}}$

We set $t_{nk}=1$ because we know that in the case where $t_{nk}=0$ the exponent will be 0, which will cause the whole relation to be 1 (which doesn't affect the product. Also, we can say that $p(t_{nk}=1 \mid \vec{\mu_k}, \vec{P_k}) = p(C_k \mid \vec{\mu_k}, \vec{P_k}) = p(C_k) = \pi_k$. Therefore, we continue with the Naive Bayes assumption:

$$\prod_{n=1}^{N} \prod_{k=1}^{K} \left[\pi_k \cdot \prod_{d=1}^{D} p(x_{nd} \mid C_k, \mu_{dk}, P_{dk}) \right]^{t_{nk}}$$

(b) For a start, we need to compute $p(x_{n1}, x_{n2}, \ldots, x_{nd} \mid \mu_{jk}, P_{jk}, C_k)$ which is the joint distribution of all the features. With Naive Bayes, we assume that they are independent, which means that we don't take into consideration potential correlation between them, we assume that there is none. In this case, we have $2D \cdot K$ parameters, as we have D variance vectors for each one of the K classes, and D mean vectors for each K class. On the other hand, if we don't assume independence we should take into account of all the potential correlations between the features (in fact, this is the most significant benefit of Naive Bayes, as this is $O(2^D)$ and by assuming independence this is reduced to $O(D)$). In this case, we get a $D \cdot D$ covariance

matrix for each of the K classes which is $D \cdot D \cdot k$ parameters, as well as $D \cdot K$ parameters for the K mean vectors of size D. Finally, since the covariance matrix is symmetric we need to subtract $(D+1)/2$ from the original D parameters. The final number of parameters thus is $D \cdot K \cdot \left(D - \frac{D+1}{2}\right) + D \cdot K$ which is equal to $\underline{DK \cdot \left(\frac{D+3}{2}\right)}$. Furthermore, an example where this assumption would lead to incorrect results, is an example where the features are correlated. Assume that we are modelling car sales in 2024 and some of the features are fuel type (electric, gas, petrol) and date that the car was released. The past years the released cars are mostly electric, in comparison to 10 years ago, therefore the features are not independent, and the assumption does not hold.

$$(c)\ \log\left(p(T, X \mid M, B)\right) = \log\left(\prod_{n=1}^{N}\prod_{k=1}^{K}\left[\pi_k \cdot \prod_{d=1}^{D} p(x_{nd} \mid c_k, \mu_{dk}, \bar{P}_{dk}\right]^{t_{nk}}\right) =$$

$$= \sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk} \cdot \left(\log \pi_k + \sum_{d=1}^{D} \log p(x_{nd} \mid c_k, \mu_{dk}, \bar{P}_{dk})\right) =$$

$$= \sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk} \cdot \left(\log \pi_k + \sum_{d=1}^{D} \log\left(\frac{\bar{P}_{dk}^{1/2}}{2\pi^{1/2}} \cdot \exp\left\{-\frac{\bar{P}_{dk}}{2}(x_{nd} - \mu_{dk})^2\right\}\right)\right) =$$

$$= \sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk} \cdot \left(\log \pi_k + \sum_{d=1}^{D}\left(\log\left(\bar{P}_{dk}^{1/2}\right) - \log\left(2\pi^{1/2}\right) - \frac{\bar{P}_{dk}}{2}(x_{nd} - \mu_{dk})^2\right)\right) =$$

This is as far as we can go so we move on with (d) and the derivation of the log likelihood relation that we ended up with.

(d) We continue and use the relation we derived in (c). We will compute its partial derivative in terms of an μ_{ij} th element and subsequently we will set it to 0 and solve for μ_{ij}.

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

$\log(\beta_{dk}^{1/2}) - \log(2\pi^{1/2})$ — discard the constants

$$= \sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk}\cdot\sum_{d=1}^{P} -\beta_{dk}\cdot\left(\frac{\partial}{\partial \mu_{ij}}(x_{nd}^2) - \frac{\partial}{\partial \mu_{ij}}(2x_{nd}\mu_{dk}) + \frac{\partial}{\partial \mu_{ij}}(\mu_{dk}^2)\right) =$$

$$= \sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk}\sum_{d=1}^{D} -\beta_{dk}\cdot\left(-2x_{nd}\delta_{id}\delta_{jk} + 2\mu_{dk}\delta_{id}\delta_{jk}\right) =$$

Remove the Kronecker deltas as well as the corresponding summations and set $\boxed{d=i}$ and $\boxed{k=j}$.

$$\sum_{n=1}^{N} t_{nj}\beta_{ij}\cdot(2x_{ni} - 2\mu_{ij}) \quad (1)$$

⊛ We have to assume that β_{ij} is not 0, because in case it was 0 then we wouldn't have a valid probability distribution. That is because if it was 0 it would imply that we have infinite variance and in any meaningful probabilistic model we expect to have at least some certainty about our data.

Set (1) equal to 0 and solve for μ_{ij}:

$$\sum_{n=1}^{N} t_{nj}\beta_{ij}(2x_{ni} - 2\mu_{ij}) = 0 \iff$$

$$\cdot\frac{(1/2)}{\neq 0} \sum_{n=1}^{N} t_{nj}\beta_{ij}\cdot x_{ni} - t_{nj}\beta_{ij}\mu_{ij} = 0 \iff$$

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

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

So the MLE of μ_{dk} is: $\boxed{\mu_{dk} = \frac{\sum_{n=1}^{N} t_{nk}\cdot x_{nd}}{\sum_{n=1}^{N} t_{nk}}}$

Which describes the following: $\sum_{n=1}^{N} t_{nk}$: The number of regions that were classified as k (a kth climate) (N_k)

$\sum_{n=1}^{N} x_{nd}\cdot t_{nk}$: The number of measurements that occurred in the kth class (N_{dk})

So we can also write the result as: $\boxed{\mu_{dk} = \frac{N_{dk}}{N_k}}$, which is the frequency of the regions that have the dth temperature measurement for the kth climate class.

(e) $p(C_1|x) \overset{(**)}{=} \frac{p(x|C_1)\cdot p(C_1)}{p(x)} \overset{\text{sum rule}}{\overset{(+)}{=}} \frac{\prod_{d=1}^{D} p(x_d|\mu_{d1},\beta_{d1})\cdot \pi_1}{\sum_{k=1}^{K} p(x|C_k)\cdot p(C_k)} =$

$= \frac{\prod_{d=1}^{D} p(x_d|\mu_{d1},\beta_{d1})\cdot \pi_1}{\sum_{k=1}^{K} \pi_k \cdot p(x|C_k)} \overset{(**)}{=} \frac{\prod_{d=1}^{D} p(x_d|\mu_{d1},\beta_{d1})\cdot \pi_1}{\sum_{k=1}^{K} \pi_k \cdot p(x|\mu_k,\beta_k)} =$

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

(+) In the second step we just marginalize over k using the sum rule in order to index μ, β, as the exercise demands a result in terms of $p(x|\mu,\beta)$

(**) In the first step as well as in the fourth step we replace C_k with μ_k, β_k as it is parameterized by these values so we can proceed with them.

(f) A generative model aims to model the joint distribution $p(x,c)$ which is the probability of both the features x and the class c. Unlike discriminative models, such as logistic regression, which directly model the conditional probability $p(c|x)$, generative models do this by factoring the joint distribution $p(x|c)\cdot p(c)$. Naive Bayes assumes that all features x_d are independent given the class c, which allows for the simplification of $p(x|c)$ as a product of individual feature likelihoods. This approach, where we first model the joint distribution and then (via Bayes' rule) we obtain $p(c|x)$ is what characterizes Naive Bayes as a generative model.

(g) We derived the following relation in (e):

$$\frac{\prod_{d=1}^{D} P(x_d \mid \mu_{d1}, \beta_{d1}) \cdot \pi_1}{\sum_{k=1}^{K} \prod_{d=1}^{D} P(x_d \mid \mu_{dk}, \beta_{dk}) \cdot \pi_k} = \frac{\prod_{d=1}^{D} \left[\frac{\beta_{d1}^{1/2}}{2\pi^{1/2}} \cdot \exp\left\{ -\frac{\beta_{d1}}{2}(x_d - \mu_{d1})^2 \right\} \right] \pi_1}{\sum_{k=1}^{K} \prod_{d=1}^{D} \left[\frac{\beta_{dk}^{1/2}}{2\pi^{1/2}} \cdot \exp\left\{ -\frac{\beta_{dk}}{2}(x_{dk} - \mu_{dk})^2 \right\} \right] \pi_k}$$

(h) The condition is:

$$p(C_1 \mid X_u) > p(C_k \mid X_u), \text{ where } k \in \mathbb{Z} \text{ and } k \in (1, K]$$

(K: total classes)

The above relation shows that the input vector x_u will be classified as "C_1", due to the probability of C_1 being higher than all of the other corresponding class probabilities.

(i) We have: $p(C_1|X_u) > p(C_k|X_u) \overset{\text{Bayes Rule}}{\Longleftrightarrow}$

$$p(C_1)\cdot p(X_u|C_1) > p(C_k)\cdot p(X_u|C_k) \quad (1)$$

We replace Ig to (1) accordingly:

$$\pi_1 \cdot \prod_{d=1}^{D} \frac{\vec{P}_{d1}^{1/2}}{2\pi^{1/2}} \exp\left\{-\frac{\vec{P}_{d1}}{2}(\vec{X}_d - \mu_{d1})^2\right\} > \pi_k \cdot \prod_{d=1}^{D} \frac{\vec{P}_{dk}^{1/2}}{2\pi^{1/2}} \exp\left\{-\frac{\vec{P}_{dk}^{1/2}}{2}(\vec{X}_d - \mu_k)^2\right\} =$$

$$= \pi_1 \prod_{d=1}^{D} \vec{P}_{d1}^{1/2} \cdot \exp\left\{-\frac{\vec{P}_{d1}}{2}(\vec{X}_d - \mu_{d1})^2\right\} > \pi_k \cdot \prod_{d=1}^{D} \vec{P}_{dk}^{1/2} \exp\left\{-\frac{\vec{P}_{dk}}{2}(\vec{X}_d - \mu_{dk})^2\right\} =$$

We log both sides:

$$= \log\pi_1 + \sum_{d=1}^{D} \log \vec{P}_{d1}^{1/2} - \frac{\vec{P}_{d1}}{2}(\vec{X}_d - \mu_{d1})^2 > \log\pi_k + \sum_{d=1}^{D} \log \vec{P}_{dk}^{1/2} - \frac{\vec{P}_{dk}}{2}(\vec{X}_d - \mu_{dk})^2 =$$

$$= \sum_{d=1}^{D} -\vec{P}_{d1}(\vec{X}_d - \mu_{d1})^2 + \vec{P}_{dk}(\vec{X}_d - \mu_{dk})^2 > 2\log\left(\frac{\pi_k}{\pi_1}\right) + \sum_{d=1}^{D} \log\frac{\vec{P}_{dk}}{\vec{P}_{d1}}$$

We set $2\log\left(\frac{\pi_k}{\pi_1}\right) + \sum_{d=1}^{D} \log\left(\frac{\vec{P}_{dk}}{\vec{P}_{d1}}\right) = C$ as these are constants and we get:

$$\sum_{d=1}^{D} \left(-\vec{P}_{d1}(\vec{X}_d - \mu_{d1})^2 + \vec{P}_{dk}(\vec{X}_d - \mu_{dk})^2\right) > C$$

We collapse the summation:

$$-\vec{P}_1(X - \vec{\mu_1}) + \vec{P}_k(X - \vec{\mu_k}) > C$$

We set $\vec{P}_1 \cdot I = B_1$, $\vec{P}_k \cdot I = B_k$

$$-B_1(X - \vec{\mu_1})^2 + B_k(X - \vec{\mu_k})^2 > C \Leftrightarrow$$

$$-(X - \vec{\mu_1})^T B_1 (X - \vec{\mu_1}) + (X - \vec{\mu_k})^T B_k (X - \vec{\mu_k}) > C \Leftrightarrow$$

$$-X^T B_1 X + X^T B_1 \vec{\mu_1} + \vec{\mu_1}^T B_1 X - \mu_1^T B_1 \mu_1 + X^T B_k X - X^T B_k \vec{\mu_k} - \vec{\mu_k}^T B_k X + \mu_k^T B_k \vec{\mu_k} > C$$

$$\Leftrightarrow X^T B_k X - X^T B_1 X - 2X^T B_k \mu_k + 2X^T B_1 \mu_1 + \mu_k^T B_k \mu_k - \mu_1^T B_1 \mu_1 > C'$$

Where $C' = C - \mu_k^T B_k \mu_k + \mu_1^T B_1 \mu_1$

So we have: $\boxed{X^T(2B_1\mu_1 - 2B_k\mu_k) + X^T(B_k - B_1)X > C'}$

Which is saying the same as (1) but written as an inequality in terms of the posterior of Ig.

(k) Since $P_{d1} = P_{dk}$ (i) becomes:

$$x^T(2B\mu_1 - 2B\mu_k) > c' \quad (1)$$

In order to prove this, we want to bring somehow the relation $x_0(1-\lambda) + \lambda x$ from (i) to (1). One simple way is to just set $x = x_0(1-\lambda) + \lambda x_1$. Then (1) transforms to:

$$(x_0(1-\lambda) + \lambda x_1)^T (2B\mu_1 - 2B\mu_k) > c'$$

We do the product and we get:

$$x_0^T(1-\lambda)\cdot(2B\mu_1 - 2B\mu_k) + \lambda x_1^T(2B\mu_1 - 2B\mu_k) > c'$$

This is a convex combination of the decision inequalities at x_0 and x_1, which accompanied by the assumption that the x_0, x_1 we selected are in the decision region C_1, prove that any point $x\lambda$ on the straight line between x_0 and x_1 also satisfies the decision boundary condition for C_1. Thus, the region is convex.

(j) If the quadratic term of (i) is convex, then the decision regions for this problem will be convex as well. The term is:

$$x^T(B_k - B_1)x \quad (1)$$

This term will be convex iff the matrix $B_k - B_1$ is positive definite, or $B_k = B_1$. The exercise hasn't specified the latter so we have to assume that it is $B_k \neq B_1$. Thus, we need to prove that $B_k - B_1$ is positive definite in order to say that it is convex. Given the nature of ~~the expression~~ matrix theory, it is very hard to be certain about the positive definiteness of $B_k - B_1$ because we are not sure that all eigenvalues will be larger than 0.

To conclude, it could be convex if we knew $B_k = B_1$ (because then we have a linear problem as the quadratic term vanishes) or in the case where $B_k - B_1$ is positive definite. Since we don't know either, we can't answer with certainty. By intuition, if $B_k \neq B_1$ it will probably be non-convex, as some eigenvalues won't be positive.

(3) (a) We start by writing down the likelihood:

$$p(T|\Phi, w_1, \dots, w_K) = \prod_{n=1}^{N} p(\vec{t_n} | \vec{\Phi_n}, w_1, \dots, w_K) \overset{t_{nk} \text{ is the same as } C_k}{=}$$

$$= \prod_{n=1}^{N} \prod_{k=1}^{K} p(C_k | \vec{\Phi_n})^{t_{nk}}$$

Then, we derive the log-likelihood:

$$\log\left(p(T|\Phi, w_1, \dots, w_K)\right) = \log\left(\prod_{n=1}^{N} \prod_{k=1}^{K} p(C_k|\vec{\Phi_n})^{t_{nk}}\right) =$$

$$= \sum_{n=1}^{N} \sum_{k=1}^{K} t_{nk} \cdot \log\left(p(C_k|\vec{\Phi_n})\right) = \sum_{n=1}^{N} \sum_{k=1}^{K} t_{nk} \cdot \log\left(y_k(\vec{\Phi_n})\right)$$

(b) $$p(w_1, \dots, w_K | \alpha) = \prod_{k=1}^{K} N(w_k | 0, \alpha^{-1} I) =$$

$$= \prod_{k=1}^{K} 2\pi^{\frac{-k}{2}} \cdot |\alpha^{-1} \cdot I|^{\frac{-1}{2}} \cdot \exp\left\{-\frac{1}{2} \cdot w_k^T \cdot (\alpha^{-1} \cdot I)^{-1} \cdot w_k\right\} =$$

$$\overset{I^{-1}=I}{=} \prod_{k=1}^{K} 2\pi^{\frac{-k}{2}} \cdot |\alpha^{-1} \cdot I| \cdot \exp\left\{-\frac{\alpha}{2} \cdot w_k^T \cdot w_k\right\}$$

Then, we compute the logarithm:

$$\log\left(p(w_1, \dots, w_K | \alpha)\right) = \log\left(\prod_{k=1}^{K} 2\pi^{\frac{-k}{2}} \cdot |\alpha^{-1} I|^{\frac{-1}{2}} \cdot \exp\left\{-\frac{\alpha}{2} \cdot w_k^T w_k\right\}\right) =$$

$$= \sum_{k=1}^{K} -\frac{k}{2} \cdot \log 2\pi - \frac{1}{2} \log(|\alpha^{-1} I|) - \frac{\alpha}{2} w_k^T w_k =$$

$$= -\frac{1}{2} \log(|\alpha^{-1} I|) + \sum_{k=1}^{K} -\frac{k}{2} \log 2\pi - \frac{\alpha}{2} w_k^T w_k$$

, which is the log-prior.

The reason why we need the logarithm is because it performs a normalization to our values and scales them in a way that is more computer friendly. For example, very low probabilities are converted to manageable negative numbers. Also, the shift from the product "Π" to the sum "Σ" can scale heavily potential small or big numbers to a manageable state.

(c) We have to write a formula for the prior.
From Bayes Rule we get:

$$p(w_1,\dots,w_K \mid \Phi, T, \alpha) = \frac{p(T \mid \Phi, w_1,\dots,w_K)\cdot p(w_1,\dots,w_K \mid \alpha)}{p(T \mid \Phi, \alpha)} =$$

$$= \frac{\left[\prod_{n=1}^{N} p(\vec{t_n} \mid \vec{\Phi_n}, w_1,\dots,w_K)\right]\cdot p(w_1,\dots,w_K \mid \alpha)}{\int p(T \mid \Phi, w_1,\dots,w_K)\cdot p(w_1,\dots,w_K \mid \alpha)\, dw_1 \dots w_K}$$

⊕ In the final step all we did was a marginalization for $w_1,\dots,w_K$.

(d) From theory we know that the maximum aposteriori estimate is calculated via the sum of the log-likelihood and the log-prior, which we then solve in order to maximize W. So we have:

$$\arg\max_w \left(\log\left(p(w_1,\dots,w_K \mid \Phi, T, \alpha)\right)\right) = \arg\max_w$$

$$= \arg\max_w \left[\prod_{n=1}^{N} p(\vec{t_n} \mid \vec{\Phi_n}, w_1,\dots,w_K)\cdot p(w_1,\dots,w_K \mid \alpha)\right]$$

$$= \arg\max_w \left[-\frac{1}{2}\cdot\log\left(|\alpha^{-1} I|\right) + \sum_{k=1}^{K}\left(-\frac{k}{2}\log 2\pi - \frac{\alpha}{2} w_k^T w_k\right) + \sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk}\cdot\log\left(y_k(\vec{\Phi_n})\right)\right]$$

remove constants

$$= \arg\max_w \left[\sum_{k=1}^{K}\left(-\frac{\alpha}{2} w_k^T w_k\right) + \sum_{n=1}^{N}\sum_{k=1}^{K}\left(t_{nk}\cdot\log\left(y_k(\vec{\Phi_n})\right)\right)\right] =$$

$w^T w = \|w\|_2^2$

$$= -\frac{\alpha}{2}\sum_{k=1}^{K}\left(\|w_k\|_2^2\right) + \sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk}\cdot\log\left(y_k(\vec{\Phi_n})\right) \quad (*)$$

(*) is identical to the expression of regularized logistic regression for K classes given by the exercise, but with opposite signs. Since we are trying to maximize (*) for W and the aim for the other expression is to minimize for W, then we can say that they are essentially solving the same problem.

$$\left(\sum_{n=1}^{N} \sum_{k=1}^{K} \left(t_{nk} \cdot \log\left(y_k(\vec{\phi}_n) \right) \right) - \left(\frac{\alpha}{2} \cdot \sum_{k=1}^{K} \left(\| w_k \|_2^2 \right) \right) \right)$$

(e) To begin with, if we are quite confident that the weights should lie around 0, it means that our prior is strong and we should increase its impact in some way. This is done by the parameter "α", which is the precision. Since the weights lie around the mean, it means that we have low variance and therefore high precision. Thus, in the expression from (d) "α" will be large and will make the rightmost term more significant than the other terms. The reason that this actually makes the weights go to 0, is that the term of the prior is essentially the L2 norm of the weights. The L2 norm has a purpose to regularize the model by reducing the weights, therefore it will add a penalty term that will increase as they grow larger. Hence, by making this term more impactful, the norm will make sure to keep the weights as low as possible and close to the mean which is 0.

(f) The MLE is:

- $\underset{w}{\text{argmax}} \left(\log \left(\prod_{n=1}^{N} p(\vec{t_n} | \vec{\phi_n}, w_1, \dots, w_K) \right) \right)$

The MAP as we saw in d is:

- $\underset{w}{\text{argmax}} \left(\log \left(\prod_{n=1}^{N} p(\vec{t_n} | \vec{\phi_n}, w_1, \dots, w_K) \right) + \log \left(p(w_1, \dots, w_K | \alpha) \right) \right)$

We can see that if the the prior term is 0 then we would have MLE = MAP. The uncertainty about the value of the weights described in the exercise, will make "α", the precision, to have a very low value, which will make the prior term (see below) to go to 0. The more uncertain we are, the more "α" decreases and thus, the more MAP approaches MLE.

$$\left(\text{Prior term}: \underset{w}{\text{argmax}} \log \left(p(w_1, \dots, w_K | \alpha) \right) = -\frac{\alpha}{2} \cdot \sum_{k=1}^{K} \| w_k \|_2^2 \right)$$

(g) We will derive the log-likelihood from (a).

$$\frac{\partial}{\partial w_j}\left(\sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk}\cdot \log\left(y_k(\vec{\phi_n})\right)\right) = \sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk}\cdot\frac{\partial}{\partial w_j}\left[\log\left(y_k(\vec{\phi_n})\right)\right] =$$

$$= \sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk}\cdot\frac{1}{y_k(\vec{\phi_n})}\cdot\left(y_k(\vec{\phi_n})\right)' \circledast$$

$\frac{\partial y_k}{\partial a_j} = y_k(I_{kj} - y_j)$ (1) ← We have this from the exercise

We want to compute $\frac{\partial y_k}{\partial w_j} \overset{\text{chain rule}}{=} \sum_p^K \left(\frac{\partial y_k}{\partial a_p}\cdot\frac{\partial a_p}{\partial w_j}\right)$ (2)

$\frac{\partial a_p}{\partial w_j} = \frac{\partial}{\partial w_j}\left(w_p^T\phi\right) = \phi^T\delta_{pj}$ (3) ← Compute the other part

(2) $\overset{(1)}{\underset{(3)}{\Rightarrow}}$ $\frac{\partial y_k}{\partial w_j} = \sum_p^K\left(y_k(I_{kj} - y_j)\cdot\phi^T\cdot\delta_{pj}\right) =$ ← Replace the derivatives

$$= \sum_p^K\left(y_k\cdot I_{kj}\cdot\phi^T\cdot\delta_{pj} - y_k\cdot y_j\cdot\phi^T\cdot\delta_{pj}\right) =$$

$$= \sum_p^K\left(y_k\cdot I_{kj}\,\phi^T\delta_{pj}\right) - \sum_p^K\left(y_k\cdot y_j\cdot\phi^T\cdot\delta_{pj}\right) =$$

$$= y_k\cdot I_{kj}\cdot\phi^T - y_k\cdot y_j\cdot\phi^T = \boxed{y_k(\vec{\phi_n})\cdot\phi^T\left(I_{kj} - y_j(\vec{\phi_n})\right)\circledast\circledast}$$

$\circledast \overset{\circledast\circledast}{\Rightarrow} \sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk}\cdot\frac{1}{y_k(\vec{\phi_n})}\cdot y_k(\vec{\phi_n})\cdot\phi^T\cdot\left(I_{kj} - y_j(\vec{\phi_n})\right) =$ ← Put it back in the initial relation and solve

$$= \sum_{n=1}^{N}\sum_{k=1}^{K}\left(t_{nk}\cdot\phi^T\cdot\left(I_{kj} - y_j(\vec{\phi_n})\right)\right) =$$

$$= \sum_{n=1}^{N}\sum_{k=1}^{K}\left(t_{nk}\cdot\phi^T\cdot I_{kj} - t_{nk}\cdot\phi^T\cdot y_j(\vec{\phi_n})\right) =$$

$$= \sum_{n=1}^{N}\sum_{k=1}^{K}\left(t_{nk}\cdot\phi^T\cdot I_{kj}\right) - \sum_{n=1}^{N}\sum_{k=1}^{K}\left(t_{nk}\cdot\phi^T\cdot y_j(\vec{\phi_n})\right) \overset{\sum_{k=1}^{K} t_{nk} = 1}{=}$$

$$= \sum_{n=1}^{N} t_{nk}\cdot\phi^T - \sum_{n=1}^{N}\phi^T\cdot y_j(\vec{\phi_n}) =$$

$$= \sum_{n=1}^{N}\phi^T\left(t_{nk} - y_j(\vec{\phi_n})\right)$$

Thus, we have computed the derivative of the log-likelihood

(4) Compute the derivative of the log prior:

$$\frac{\partial}{\partial w_j}\left(\log\left(p(w_1,\dots,w_K|\alpha)\right)\right) =$$

$$= \frac{\partial}{\partial w_j}\left[-\frac{1}{2}\left(\log(|\alpha^{-1}I|) + \sum_{k=1}^{K} k\cdot\log 2\pi + \alpha\, w_k^T w_k\right)\right] =$$

$$= \frac{\partial}{\partial w_j}\left(\sum_{k=1}^{K} -\frac{\alpha}{2} w_k^T w_k\right) = -\frac{\alpha}{2}\sum_{k=1}^{K}\frac{\partial}{\partial w_j}\left(w_k^T w_k\right) =$$

$$= -\frac{\alpha}{2}\sum_{k=1}^{K} 2 w_k \cdot \delta_{kj} = -\frac{\alpha}{2}\cdot 2 w_j = -\alpha\cdot w_j$$

Compute the derivative of the log posterior:

$$\frac{\partial}{\partial w_j}\left(\log\left(\prod_{n=1}^{N} p(\vec{t_n}|\vec{\varphi_n}, w_1,\dots,w_K)\right) + \log\left(p(w_1,\dots,w_K|\alpha)\right)\right) =$$

$$= \frac{\partial}{\partial w_j}\log\left(\prod_{n=1}^{N} p(\vec{t_n}|\vec{\varphi_n}, w_1,\dots,w_K)\right) + \frac{\partial}{\partial w_j}\log\left(p(w_1,\dots,w_K|\alpha)\right) =$$

$$\overset{\text{substitute gradients}}{=} \sum_{n=1}^{N}\varphi^T(t_{nk} - y_j(\vec{\varphi_n})) - \alpha\, w_j$$

Hence, we have computed the gradient for the log-prior and the log-posterior.

(2) (a)(i) Logistic regression with linear features tries to find a straight line decision boundary that best separates the classes. By observing the dataset we can say that it is not a linearly separable dataset, which means that a straight line as a decision boundary would be prone to misclassifications. Thus, logistic regression with linear features won't be able to correctly classify the dataset.

(ii) Logistic regression with non-linear basis functions extends the capabilities of the basic logistic regression we saw previously. If we assign non-linear basis functions, logistic regression will be able to classify the dataset correctly. What we essentially want is an ellipse-like shape to maintain inside it one class and everything outside of it will belong to the other class. I believe that if we add quadratic basis functions, logistic regression will be able to "draw" this ellipsoid.

(iii) An MLP with 1-hidden layer as we can see in Bishop 4.17. can only "draw" straight lines. However, if we use a non-linear activation function then an MLP can approximate any decision boundary, which is exactly what makes it so powerful. The MLP will be able to make this classification effectively and correctly. A potential choice for an activation function is the sigmoid function (which obviously is non-linear).

(b) We consider the subset $\{(-1,1),(1,1),(-1,-1),(1,-1)\}$ and we will prove that its not possible to classify this using Naive Bayes.

Let's take the point $(1,1)$ and classify it as C_1. This means that:

- $p(C_1 \mid X_1=1, X_2=1) > p(C_2 \mid X_1=1, X_2=1)$ (1)

We repeat this for the other 3 points as well:

- $p(C_1 \mid X_1=-1, X_2=-1) > p(C_2 \mid X_1=-1, X_2=-1)$ (2)
- $p(C_1 \mid X_1=-1, X_2=1) < p(C_2 \mid X_1=-1, X_2=1)$ (3)
- $p(C_1 \mid X_1=1, X_2=-1) < p(C_2 \mid X_1=-1, X_2=-1)$ (4)

We classified each point according to the shape, separating the classes "in diagonals" (s.t. each class is assigned 2 points in a non-linearly separable manner)

Then, we apply the Bayes rule as well as the Naive Bayes assumption to each relation.

(1) $\Rightarrow p(X_1=1|C_1)\cdot p(X_2=1|C_1)\cdot p(C_1) > p(X_1=1|C_2)\cdot p(X_2=1|C_2)\cdot p(C_2)$

$\Leftrightarrow \frac{p(X_1=1|C_1)\cdot p(X_2=1|C_1)\cdot p(C_1)}{p(X_1=1|C_2)\cdot p(X_2=1|C_2)\cdot p(C_2)} > 1$ (5)

(2) $\Rightarrow p(X_1=-1|C_1)\cdot p(X_2=-1|C_1)\cdot p(C_1) > p(X_1=-1|C_2)\cdot p(X_2=-1|C_2)\cdot p(C_2)$

$\Leftrightarrow \frac{p(X_1=-1|C_1)\cdot p(X_2=-1|C_1)\cdot p(C_1)}{p(X_1=-1|C_2)\cdot p(X_2=-1|C_2)\cdot p(C_2)} > 1$ (6)

(3) $\Rightarrow p(X_1=-1|C_1)\cdot p(X_2=1|C_1)\cdot p(C_1) < p(X_1=-1|C_2)\cdot p(X_2=1|C_2)\cdot p(C_2)$

$\Leftrightarrow \frac{p(X_1=-1|C_2)\cdot p(X_2=1|C_2)\cdot p(C_2)}{p(X_1=-1|C_1)\cdot p(X_2=1|C_1)\cdot p(C_1)} > 1$ (7)

(4) $\Rightarrow p(X_1=1|C_1)\cdot p(X_2=-1|C_1)\cdot p(C_1) < p(X_1=1|C_2)\cdot p(X_2=-1|C_2)\cdot p(C_2)$

$\Leftrightarrow \frac{p(X_1=1|C_2)\cdot p(X_2=-1|C_2)\cdot p(C_2)}{p(X_1=1|C_1)\cdot p(X_2=-1|C_1)\cdot p(C_1)} > 1$ (8)

Subsequently, we multiply (5) with (7) and (6) with (8):

(5) $\times$ (7) $\Rightarrow \frac{p(X_1=1|C_1)\cdot p(X_2=1|C_1)\cdot p(C_1)\cdot p(X_1=-1|C_2)\cdot p(X_2=1|C_2)\cdot p(C_2)}{p(X_1=1|C_2)\cdot p(X_2=1|C_2)\cdot p(C_2)\cdot p(X_1=-1|C_1)\cdot p(X_2=1|C_1)\cdot p(C_1)} > 1$

$\Leftrightarrow \frac{p(X_1=1|C_1)\cdot p(X_1=-1|C_2)}{p(X_1=1|C_2)\cdot p(X_1=-1|C_1)} > 1$ (9)

(6) $\times$ (8) $\Rightarrow \frac{p(X_1=-1|C_1)\cdot p(X_2=-1|C_1)\cdot p(C_1)\cdot p(X_1=1|C_2)\cdot p(X_2=-1|C_2)\cdot p(C_2)}{p(X_1=-1|C_2)\cdot p(X_2=-1|C_2)\cdot p(C_2)\cdot p(X_1=1|C_1)\cdot p(X_2=-1|C_1)\cdot p(C_1)} > 1$

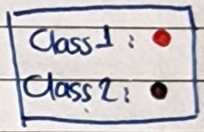
$\Leftrightarrow \frac{p(X_1=-1|C_1)\cdot p(X_1=1|C_2)}{p(X_1=-1|C_2)\cdot p(X_1=1|C_1)} > 1$ (10)

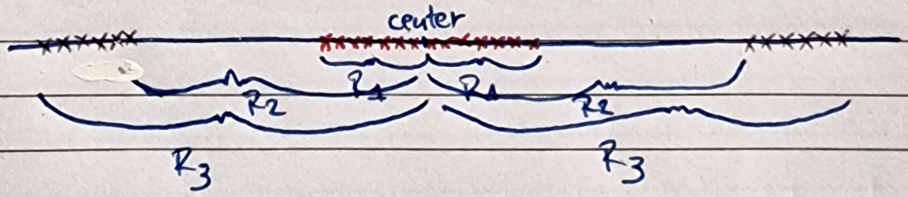
We observe that (9) and (10) are reciprocals and it's impossible for them to be greater than 1. Therefore, we have arrived at a contradiction. The only assumption that we made throughout this whole proof was the Naive Bayes assumption. Thus, we have proved that Naive Bayes can not classify this dataset.

CamScanner

(c) The reasoning for this exercise will be different than (2b), because in contrast with that question, I believe that Naive Bayes can classify this dataset.

→ The dataset given in these ~~two~~ exercises causes 2 significant problems that prevent Naive Bayes from working. First, the classes are obviously non-linearly separable and second, there is intense correlation between x_1 and x_2. I suggest that we switch the coordinates (x, y) to polar coordinates (r, θ) where it will be much simpler to separate linearly the classes. This transformation solves both issues, as in polar coordinates we will take advantage of the center of the data and the way they are circular around it (the orange data are a circular uniform distribution, while the other class is called annulus in statistics), and we also solve the correlation issue, as the radius "r" tells us nothing about the angle "θ" and vice versa. Also, we choose to discard the feature θ altogether, as it won't contribute in any way to the classification and performing Naive Bayes in one dimension with only the radius will be much simpler. This scheme is depicted below:





With the above implementation, every point with radius up to R_1 will be classified as 'class 1', while all the rest can be classified as 'class 2'. In fact, this implementation can work for more than 2 classes by taking advantage of the other radiuses as well, as long as they maintain the same center and the same uniform formation such as the classes that we have.

(d) The main difference between logistic regression with non-linear basis functions Φ, and multilayer perceptrons in terms of Φ, is that we manually choose static non-linear basis functions for logistic regression, while in the case of MLP's, the functions are determined by the model weights that are updated in every epoch. At first the MLP chooses random weights that set random basis functions. In each epoch the weights are adjusted and then they adjust the basis functions. This is the main difference with logistic regression, as there we have to select the non-linear basis functions beforehand and they don't change during the process. In fact, this is what makes MLPs so attractive, as it is not always easy (or even) to know in advance the proper basis functions.