Link to code: [Github]

Q1

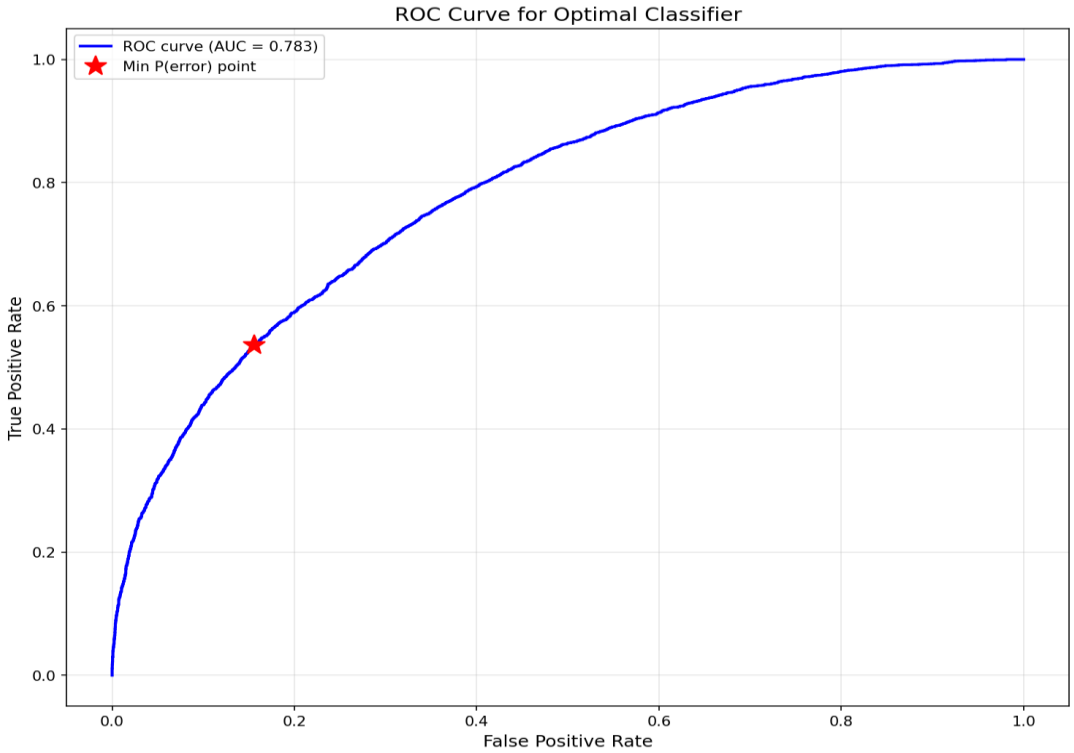
Part 1

$$p(L = 0/x) = \frac{p(x/L = 0)p(L = 0)}{p(x)}$$

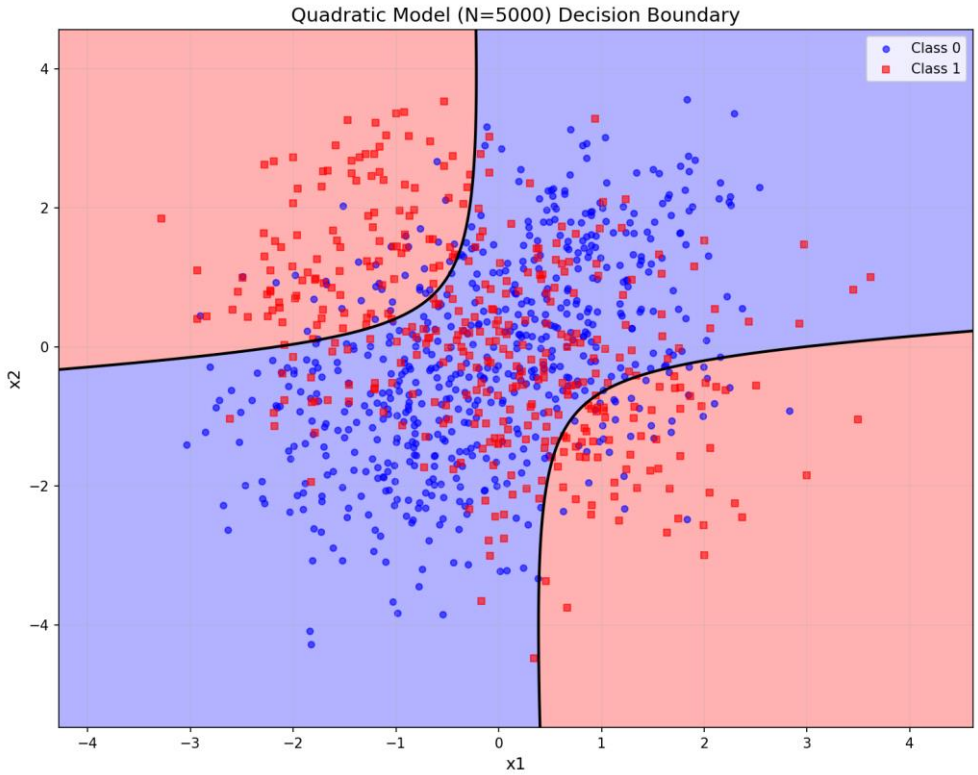
$$p(L = 1/x) = \frac{p(x/L = 1)p(L = 1)}{p(x)}$$

$$If\ p(L = 0/x)\ >\ p(L = 1/x) \longrightarrow\ Class\ 0$$

$$p(L = 0/x)\ <\ p(L = 1/x) \longrightarrow\ Class\ 1$$



ROC curve for optimal Bayes classifier on 10,000 validation samples with area under the curve=0.783. The red star marks the min-P(error)=0.2807 when P(L=1|x) = 0.5 = P(L=0|x). The linear model performed significantly worse, with an error of 0.5151 at N=50 and 0.4047 at N=5000, though it is similar to the error of 0.4003 at N=500. Quadratic model performed comparably to Bayesian classifier error of 0.2988 at N=50 and an error of 0.2855 at N=5000. The linear model performs worse due to wrong model selection, whereas the quadratic model captures complexity sufficiently.



From the decision boundary above, there is significant class overlap visible in the central region where the four Gaussian components intersect.

Q2

$\boldsymbol{y = c(x,w) + v}$ and $\boldsymbol{v \sim N(0, \sigma^2)}$

$$c(x,w) = w_0 + w_1x_1 + w_2x_2 + w_3x_1^2 + w_4x_1x_2 + w_5x_2^2 + w_6x_1^3 + w_7x_1^2x_2 + w_8x_1x_2^2 + w_9x_2^3$$
$$= w^T z(x)$$

Where z(x)=$[1, x_1, x_2, x_1^2, x_2^2, x_1x_2, x_2^2, x_1^3, x_1^2x_2, x_1x_2^2, x_2^3]^T$

For ML estimator

D = $[(x_1, y_1), (x_2, y_2), \ldots\ldots, (x_n, y_n)]$

$$\mathbf{p(D|w, \sigma^2)} = \prod_{n=1}^{N} \left(2\pi\sigma^2\right)^{-1/2} e^{-\frac{1}{2}(y_n - c(x_n,w))^T \sigma^{-2} (y_n - c(x_n,w))}$$

$$\mathbf{J(w)} = ln(\mathbf{p}(D|w, \sigma^2)) = -\frac{1}{2\sigma^2} \sum_{n=1}^{N} (y_n - c(x_n, w))^T (y_n - c(x_n, w))$$

$$\mathbf{J(w)} = (Y - Zw)^T (Y - Zw)$$

$$\Rightarrow (Y^T - w^T Z^T)(Y - Zw) = Y^T Y - Y^T Zw - w^T Z^T Y + w^T Z^T Zw$$

$$\mathbf{J(w)} = Y^T Y - 2w^T Z^T Y + w^T Z^T Zw$$

$$\frac{\partial J(w)}{\partial w} = -2Z^T Y + 2Z^T Z w = 0$$

$$\hat{w}_{ML} = (Z^T Z)^{-1} Z^T Y$$

For MAP estimation

$$p(\mathbf{w}|\mathbf{D}) = p(D|w)p(w) \; and \; w \sim N(0, \gamma I)$$

$$ln(p(\mathbf{w}|\mathbf{D})) = ln(p(D|w)) + ln(p(w))$$

$$p(w) = (2\pi)^{-d/2} |\gamma I|^{-\frac{1}{2}} e^{-\frac{1}{2} w^T \gamma^{-1} w}$$

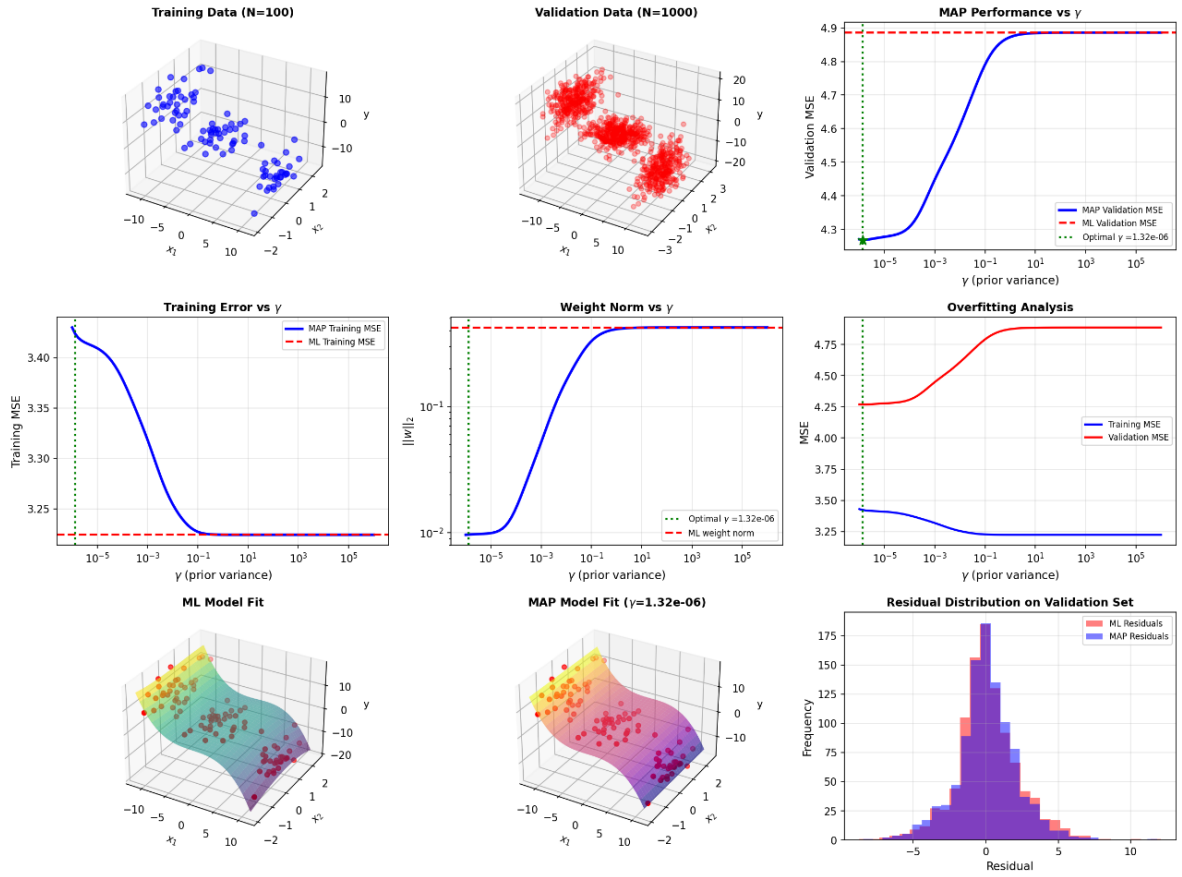
$$\ln(\mathbf{p}(\mathbf{w})) = -\frac{1}{2\gamma} w^T w$$

$$min\big(J(w)\big) = -\frac{1}{2\gamma} w^T w - \frac{1}{2\sigma^2} (Y - Zw)^T (Y - Zw)$$

$$\frac{\partial J(w)}{\partial w} = -\frac{2w}{2\gamma} - \frac{2Z^T Z w - 2Z^T Y}{2\sigma^2}$$

$$\hat{w}_{MAP} = (\frac{\sigma^2}{\gamma} I + Z^T Z)^{-1} Z^T Y$$

We can observe that as $\lim_{\gamma \to \infty} \hat{w}_{MAP} = \hat{w}_{ML}$



The ML estimator results in overfitting, with a Validation MSE of 4.8862. This is because there are only 100 samples for training and 1000 samples for validation. MAP performs better than the ML estimator due to the regularisation term $\left(\frac{\sigma^2}{\gamma} \boldsymbol{I}\right)$. Best γ: 1.3219e-06 and MAP estimator MSE with validation data was 4.2684. Which resulted in an improvement of 12.64% compared to ML estimator

Q3

$$\mathbf{P}\left(\begin{bmatrix}\mathbf{x}\\ \mathbf{y}\end{bmatrix}\right) = \frac{\mathbf{1}}{\mathbf{2\pi\sigma_x\sigma_y}} \mathbf{exp} - \frac{\mathbf{1}}{\mathbf{2}} \begin{bmatrix}\mathbf{x}\\ \mathbf{y}\end{bmatrix} \begin{bmatrix}\boldsymbol{\sigma_x^2} & \mathbf{0}\\ \mathbf{0} & \boldsymbol{\sigma_y^2}\end{bmatrix}^{-\mathbf{1}} \begin{bmatrix}\mathbf{x}\\ \mathbf{y}\end{bmatrix}$$

$$\mathbf{p}(\mathbf{r_i}|\mathbf{x},\mathbf{y}) = \left(\mathbf{2\pi\sigma_i^2}\right)^{-\mathbf{1}/\mathbf{2}} \mathbf{exp}\left[-\frac{(\mathbf{r_i}-\mathbf{d_i})^2}{\mathbf{2\sigma_i^2}}\right]$$

Where $\mathbf{d_i} = \sqrt{(\boldsymbol{x}-\mathbf{x_i})^2 + (\mathbf{y}-\mathbf{y_i})^2}$

$$\mathbf{p}(\mathbf{x},\boldsymbol{y}|\boldsymbol{r}) = \mathbf{p}\left(\mathbf{r}|\mathbf{x},\mathbf{y}\right)\mathbf{p}(\mathbf{x},\mathbf{y})$$

$$\mathbf{ln}(\,\mathbf{p}(\mathbf{x},\boldsymbol{y})) \;=\; -\frac{\mathbf{1}}{\mathbf{2}}\left\{\frac{\mathbf{x^2}}{\boldsymbol{\sigma_x^2}} + \frac{\mathbf{y^2}}{\boldsymbol{\sigma_y^2}}\right\}$$

$$\mathbf{ln}(\,\mathbf{p}(\mathbf{r}|\mathbf{x},\mathbf{y})) \;=\; -\sum \frac{(\mathbf{r_i}-\mathbf{d_i})^2}{\mathbf{2\sigma_i^2}}$$

$$\mathbf{ln}(\,\mathbf{p}(\mathbf{x},\boldsymbol{y}|\boldsymbol{r})) \;=\; -\sum \frac{(\mathbf{r_i}-\mathbf{d_i})^2}{\mathbf{2\sigma_i^2}} - \frac{\mathbf{1}}{\mathbf{2}}\left[\frac{\mathbf{x^2}}{\boldsymbol{\sigma_x^2}} + \frac{\mathbf{y^2}}{\boldsymbol{\sigma_y^2}}\right]$$

For MAP estimation $[\mathbf{x_{MAP}}, \mathbf{y_{MAP}}]^{\boldsymbol{T}} = \boldsymbol{min}(\mathbf{ln}(\,\mathbf{p}(\boldsymbol{x},\boldsymbol{y}|\boldsymbol{r})))$

$$= \boldsymbol{min}\left(-\sum \frac{(\mathbf{r_i}-\mathbf{d_i})^2}{\mathbf{2\sigma_i^2}} - \frac{\mathbf{1}}{\mathbf{2}}\left[\frac{\mathbf{x^2}}{\boldsymbol{\sigma_x^2}} + \frac{\mathbf{y^2}}{\boldsymbol{\sigma_y^2}}\right]\right)$$

The true position of the vehicle is set at $[0.3, 0.4]^T$, noise variance of $\sigma_i = 0.3$ and $\sigma_x = \sigma_y = 0.25$. The code generates evenly spaced K landmarks on a unit circle. Noise data was generated. Then the log likelihood is set up according to the equation above. To get $\mathbf{[x_{MAP}, y_{MAP}]}^{\boldsymbol{T}}$ we have to minimize the above equation.

K = 1 Landmarks (on unit circle): Landmark 1: [1.000, 0.000]

r1 = 0.9552 (true distance = 0.8062, noise_error = +0.1490)

MAP estimate: [0.0183, -0.0000], Localization error: 0.4892

K = 2 Landmarks (on unit circle): Landmark 1: [1.000, 0.000] Landmark 2: [-1.000, 0.000]

r1 = 0.7647 (true distance = 0.8062, noise_error = -0.0415)

r2 = 1.5545 (true distance = 1.3601, noise_error = +0.1943)

MAP estimate: [0.2296, -0.0000], Localization error: 0.4062

K = 3 Landmarks (on unit circle): Landmark 1: [1.000, 0.000] Landmark 2: [-0.500, 0.866] Landmark 3: [-0.500,-0.866]

r1 = 1.2631 (true distance = 0.8062, noise_error = +0.4569)

r2 = 0.8556 (true distance = 0.9258, noise_error= -0.0702)

r3 = 1.4274 (true distance = 1.4976, noise_error = -0.0702)

MAP estimate: [-0.0457, 0.1840], Localization error: 0.4077

K = 4 Landmarks (on unit circle): Landmark 1: [1.000, 0.000] Landmark 2: [0.000, 1.000] Landmark 3: [-1.000, 0.000] Landmark 4: [-0.000,-1.000]

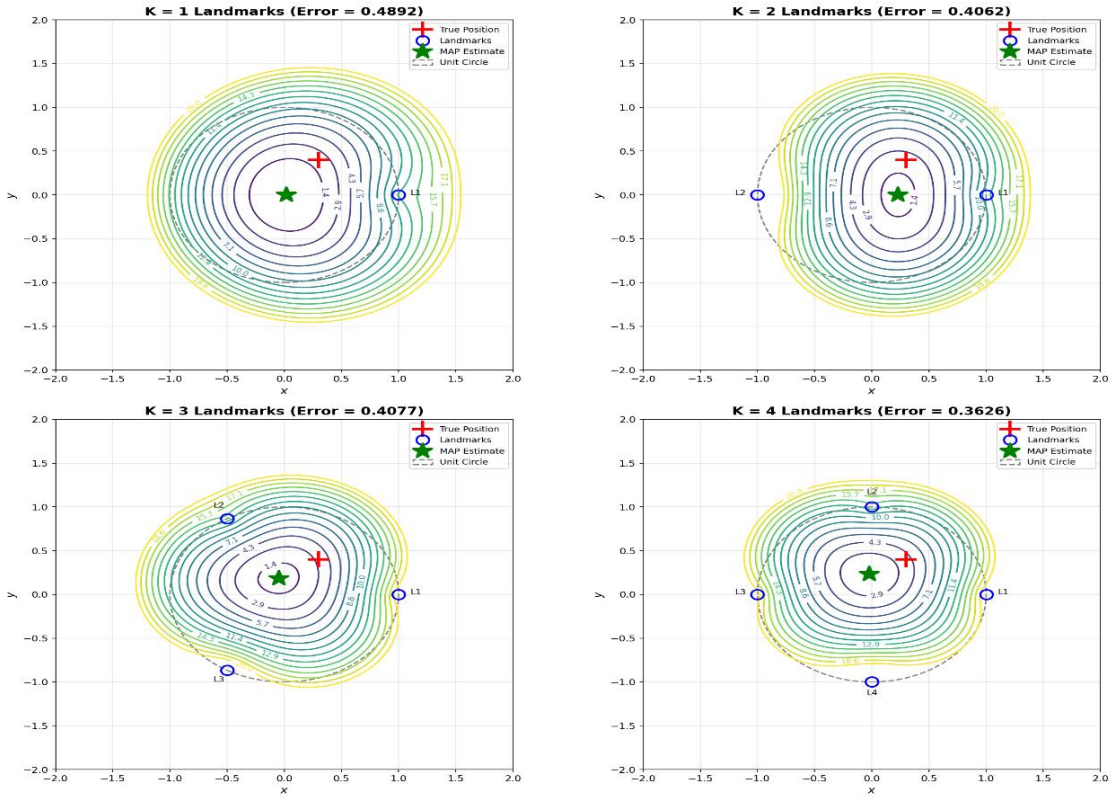
r1 = 1.2800 (true distance = 0.8062, noise_error= +0.4738)

r2 = 0.9011 (true distance = 0.6708, noise_error = +0.2302)

r3 = 1.2193 (true distance = 1.3601, noise_error = -0.1408)

r4 = 1.5946 (true distance = 1.4318, noise_error = +0.1628)

MAP estimate: [-0.0207, 0.2308], Localization error: 0.3626



We observe that as the number of K increases, localization error decreases. As K increases, the contours gradually become circle-shaped, centred around the estimated position. The numbers represent the value of the objective function for each (x,y) value. For K=1, the innermost visible contours (J = 2.9) cover a large area, indicating the minimum is spread over a broad, shallow region. For K=4, the innermost contours (J ≈ 2.9) cover a much smaller area, which means higher certainty.

Q4

$$\lambda(\alpha_i, w_j) = \begin{cases} 0, i = j = 1,2,\dots,c \\ \lambda_r\,, c+1 \\ \lambda_s, otherwise \end{cases}$$

$$R(\alpha_i, x) \;=\; \sum_{j=1}^{c} \lambda_s(\alpha_i/w_j)\,\mathbf{p}(\mathbf{w_j}\,/\,\mathbf{x})$$

$$= \lambda_s \sum_{i\neq j}^{c} \mathbf{p}(\mathbf{w_j}\,/\,\mathbf{x}) \;=\; \lambda_s(1 - \mathbf{p}(\mathbf{w_i}\,/\,\mathbf{x}))$$

$$R(rejection/x) \;=\; \lambda_r$$

If we decide that $\mathbf{w_i}$ class is better than all the other classes, the risk for class $\mathbf{w_i}$ should be the least among all classes and risk of $\mathbf{w_i}$ should be less than rejection

Condition 1. $\mathbf{R(\alpha_i/x) \leq R(\alpha_j/x)}$

$$\lambda_s(1 - \mathrm{p(w_i\,/\,x)}) \leq\; \lambda_s(1 - \mathrm{p(w_j\,/\,x)})$$

$$\mathrm{p(w_i\,/\,x)} \geq \mathrm{p(w_j\,/\,x)}$$

and **Condition 2.** $\mathbf{R(\alpha_i/x) < R(rejection/x)}$

$$\lambda_s(1 - \mathrm{p(w_i\,/\,x)}) < \lambda_r$$

$$\mathrm{p(w_i\,/\,x)} > 1 - \frac{\lambda_r}{\lambda_s}$$

Otherwise reject

Special Case 1.

$\boldsymbol{\lambda_r = 0}$

This makes rejection free of loss which in turn makes **Condition2** $\mathbf{p(w_i\,/\,x) > 1}$ which is not possible as all probabilities are ≤ 1. This makes the classifier always rejecting

Special Case 2.

$\boldsymbol{\lambda_r > \lambda_s}$

$$\mathbf{1 - \frac{\lambda_r}{\lambda_s} < 0}$$

Condition 2 now becomes always satisfied as all probabilities are between 0 and 1. This makes classifier to never reject instead always classifies data to most likely class

Q5

ML estimation for Θ

$$\mathbf{p}(\mathbf{D}|\boldsymbol{\Theta}) = \prod_{n=1}^{N} \boldsymbol{p}(\boldsymbol{z_n}|\boldsymbol{\theta})$$

$$\mathbf{p}(\mathrm{z}|\boldsymbol{\Theta}) = \prod_{k=1}^{K} \boldsymbol{\theta_k^{z_k}}$$

$$\mathbf{p}(\mathrm{D}|\boldsymbol{\Theta}) = \prod \prod_{k=1}^{K} \prod_{n=1}^{N} \boldsymbol{\theta_k^{z_{n_k}}} = \prod_{k=1}^{K} \boldsymbol{\theta_k^{N_k}} \text{ here } \boldsymbol{N_k} = \sum_{n=1}^{N} \boldsymbol{z_{n_k}}$$

$$\mathbf{ln}(\,\mathbf{p}(\mathrm{D}|\boldsymbol{\Theta})) = \sum_{k=1}^{K} \boldsymbol{N_k}\,\mathbf{ln}(\,\boldsymbol{\theta_k}) \;\; \boldsymbol{and\ constraint} \;\; \sum_{k=1}^{K} \boldsymbol{\theta_k} = \mathbf{1}$$

Using Lagrange multiplier λ

$$\mathcal{L}(\boldsymbol{\theta}, \lambda) = \sum_{k=1}^{K} \boldsymbol{N_k}\,\mathbf{ln}(\,\boldsymbol{\theta_k}) + \lambda(1 - \sum_{k=1}^{K} \boldsymbol{\theta_k})$$

$$\frac{\partial \mathcal{L}}{\partial \theta_k} = \frac{\boldsymbol{N_k}}{\boldsymbol{\theta_k}} - \lambda = 0$$

$$\boldsymbol{\theta_k} = \frac{\boldsymbol{N_k}}{\lambda}$$

$$\sum_{k=1}^{K} \boldsymbol{\theta_k} = \mathbf{1} = \sum_{k=1}^{K} \frac{\boldsymbol{N_k}}{\lambda}$$

$$\widehat{\boldsymbol{\theta}}_{\boldsymbol{ML}} = \frac{\boldsymbol{N_k}}{\sum_{k=1}^{K} \boldsymbol{N_k}}$$

If p(Θ) has a Dirichlet distribution

$$\mathrm{p}(\Theta|\alpha) = \frac{1}{B(\alpha)} \prod \prod_{k=1}^{K} \boldsymbol{\theta_k^{\alpha_k - 1}}$$

$$\mathbf{p}(\boldsymbol{\Theta}|\mathbf{D}, \alpha) = \mathbf{p}(\mathbf{D}|\boldsymbol{\Theta}, \alpha)\mathrm{p}(\Theta|\alpha)$$

$$= \prod \prod_{k=1}^{K} \boldsymbol{\theta_k^{N_k}} \prod \prod_{k=1}^{K} \boldsymbol{\theta_k^{\alpha_k - 1}}$$

$$= \prod_{k=1}^{K} \boldsymbol{\theta_k^{N_k + \alpha_k - 1}}$$

$$\ln(\mathbf{p}(\Theta|\boldsymbol{D},\alpha)) = \sum_{k=1}^{K}(\boldsymbol{N_k}+\alpha_k-\mathbf{1})\ln(\boldsymbol{\theta_k}) \textbf{ and constraint } \sum_{k=1}^{K}\boldsymbol{\theta_k}=\mathbf{1}$$

$$\mathcal{L}(\boldsymbol{\theta},\lambda) = \sum_{k=1}^{K}(\boldsymbol{N_k}+\alpha_k-\mathbf{1})\ln(\boldsymbol{\theta_k}) + \lambda\left(1-\sum_{k=1}^{K}\boldsymbol{\theta_k}\right)$$

$$\frac{\partial\mathcal{L}}{\partial\theta_k} = \frac{(\boldsymbol{N_k}+\alpha_k-\mathbf{1})}{\boldsymbol{\theta_k}} - \lambda = 0$$

$$\boldsymbol{\theta_k} = \frac{(\boldsymbol{N_k}+\alpha_k-\mathbf{1})}{\lambda}$$

$$\sum_{k=1}^{K}\boldsymbol{\theta_k} = \mathbf{1} = \sum_{k=1}^{K}\frac{(\boldsymbol{N_k}+\alpha_k-\mathbf{1})}{\lambda}$$

$$\lambda = \sum_{k=1}^{K}(\boldsymbol{N_k}+\alpha_k-\mathbf{1})$$

$$\widehat{\boldsymbol{\theta}}_{\boldsymbol{MAP}} = \frac{(\boldsymbol{N_k}+\alpha_k-\mathbf{1})}{\sum_{k=1}^{K}(\boldsymbol{N_k}+\alpha_k-\mathbf{1})}$$

References

https://scikit-learn.org/stable/modules/preprocessing.html#polynomial-features