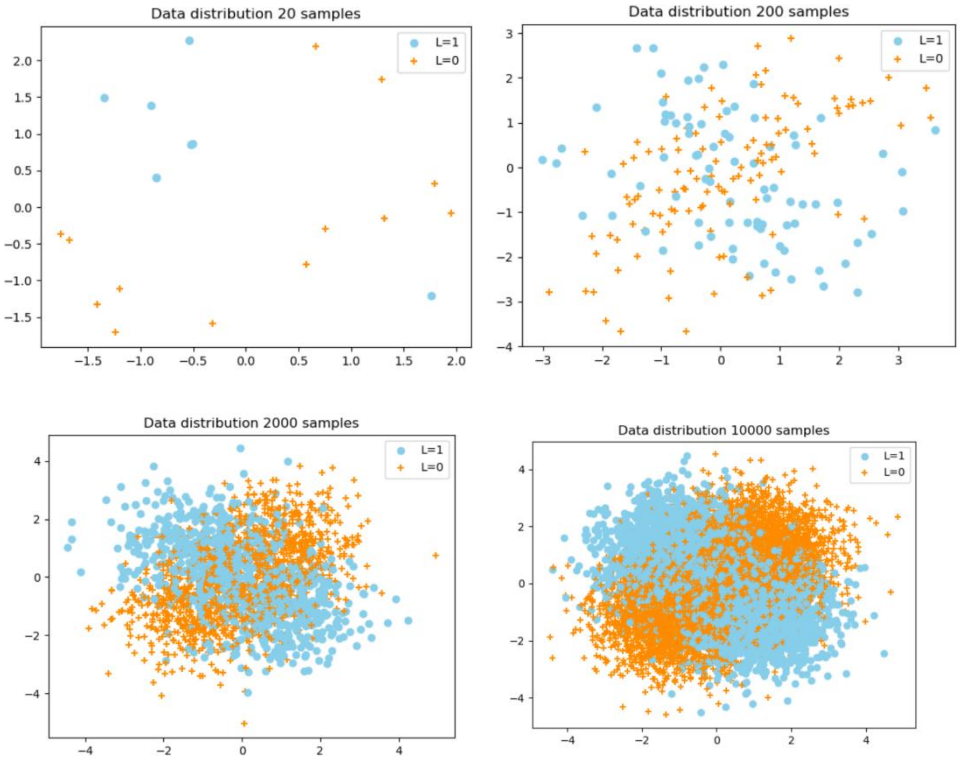
Question 1

Part 1.

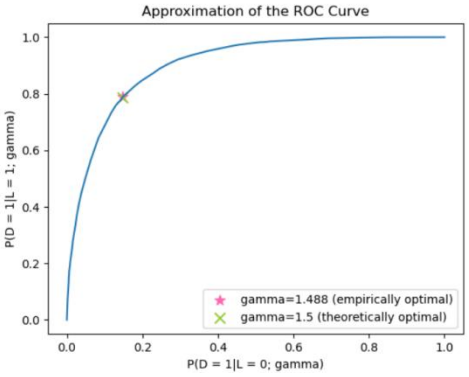
The data distributions of the generated datasets are as below:



The theoretically optimal classifier that achieves minimum probability of error is the ERM classifier with 0-1 loss, or the Maximum a Posteriori (MAP) classifier. Therefore, the decision rule for a two-class ERM classifier can be expressed as the following likelihood-ratio test:

$$\frac{p(\mathbf{x} \mid L=1)}{p(\mathbf{x} \mid L=0)} >^{?} \gamma = \frac{p(L=0)}{p(L=1)} \frac{\lambda_{10} - \lambda_{00}}{\lambda_{01} - \lambda_{11}}$$

Since $\lambda_{10}=\lambda_{01}=1$, $\lambda_{11}=\lambda_{00}=0$. Thus we can conclude that if $\frac{p(x|L=1)}{p(x|L=0)} > \frac{0.6}{0.4} = 1.5$, the optimal the optimal classifier will label it as L=1, otherwise it will be labeled as L=0.I applied the classifier to containing 10k samples and generated its corresponding ROC curve indicating the theoretical and the estimated min-error threshold is estimated and plotted as follow



:

	Threshold	min{P(error)}
Theoretically Calculated	1.500	0.1737
Estimated	1.488	0.1735

It can be observed that as the validation dataset size increases, the theoretically calculated threshold and the estimated threshold for minimizing the probability of error become increasingly similar.

Part 2.

For linear regression, the regression function is $h(x, w) = \frac{1}{1+e^{-w^T z(x)}}$, where $z(x) = [1, x^T]^T$.

For quadratic regression, the regression function is $h(x, w) = \frac{1}{1+e^{-w^T z(x)}}$, where $z(x) = [1, x_1, x_2, x_1^2, x_1, x_2, x_2^2]^T$.The $h(x, w)$ indicates the estimation of $p(Y=1|X)$ using the sigmoid function. Let y_n denotes the determined label of n^{th} the training sample x_n. To choose the optimal parameter w for regression, we need to specify the optimization problem as minimization of the negative-log-likelihood of the training dataset:

$$\begin{aligned}\mathbf{w} &= \underset{\omega}{argmin}(-\sum_{n=1}^{N} \ln P(y_n|\mathbf{z}(\mathbf{x}_n), \mathbf{w})) \\ &= \underset{\omega}{argmin}(-\sum_{n=1}^{N} (y_n \ln P(y_n = 1|\mathbf{z}(\mathbf{x}_n), \mathbf{w}) + (1 - y_n) \ln P(y_n = 0|\mathbf{z}(\mathbf{x}_n), \mathbf{w}))) \\ &= \underset{\omega}{argmin}(-\sum_{n=1}^{N} (y_n \ln h(\mathbf{x}_n, \mathbf{w}) + (1 - y_n) \ln(1 - h(\mathbf{x}_n, \mathbf{w}))))\end{aligned}$$

To approach the optimization,With the estimated class label posteriors, we implement a new $min\{P(error)\}$ classifier by calculate:

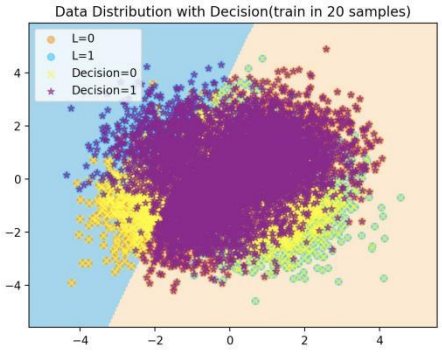
$$\gamma = \ln \frac{P(y_n = 1|\mathbf{z}(\mathbf{x}), \mathbf{w})}{P(y_n = 0|\mathbf{z}(\mathbf{x}), \mathbf{w})} = \mathbf{w}^T \mathbf{z}(\mathbf{x})$$

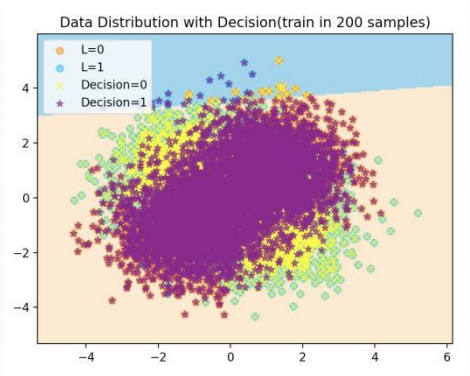
Calculate the approximation of $P(error)$ over the dataset:

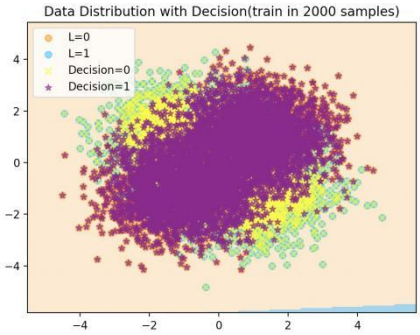
training dataset	Linear Regression	Quadratic Regression
D^{20}	0.3392	0.3321
D^{200}	0.4015	0.2841
D^{2000}	0.3927	0.2486

The plots of the decision boundary of each classifier training on different dataset are as follow

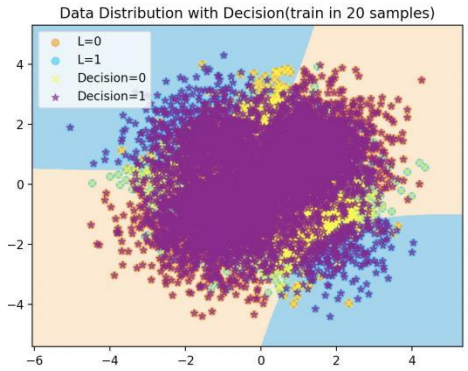
For linear regression:

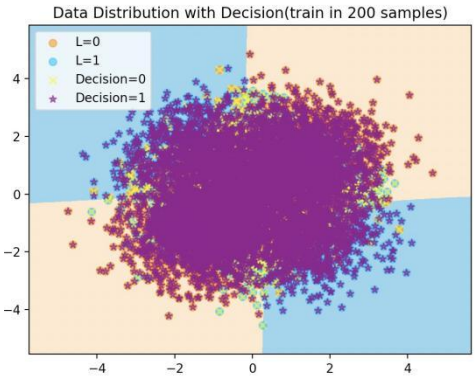


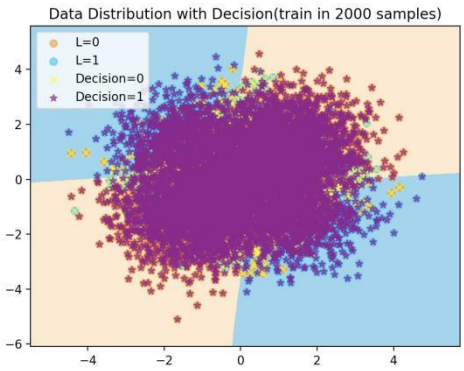




For Quadratic regression:



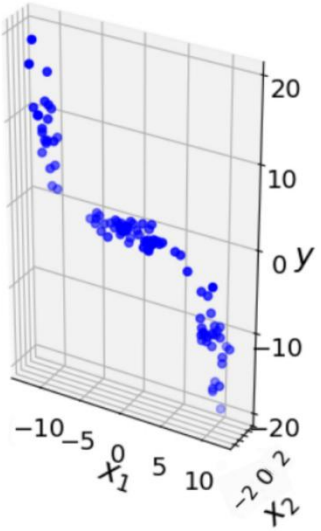




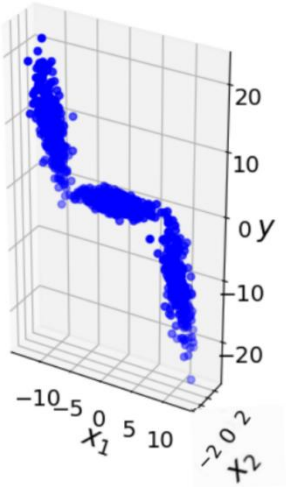
It is evident that linear regression does not fit our data distribution adequately, resulting in a high probability of error. On the other hand, the classifier trained using the quadratic regression method performs significantly better than linear regression. With quadratic regression, we observe that the

probability of error decreases with an increase in the size of the training dataset. This suggests that increasing the number of samples leads to improved modeling, which ultimately results in better performance.

Question 2



plot 2.1 Generate trainset data distribution with 100 samples



Plot 2.2 Generate trainset data distribution with 1000 samples

ML Estimation:

Defining the augmented features $\tilde{x}$ for a two-dimensional real input vector and the corresponding regression weights w after applying a cubic polynomial c(x,w) transformation:

$$\tilde{\mathbf{x}} = \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ x_1^2 \\ x_1 x_2 \\ x_2^2 \\ x_1^3 \\ x_1^2 x_2 \\ x_1 x_2^2 \\ x_2^3 \end{bmatrix} \in \mathbb{R}^{10}, \qquad \mathbf{w} = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ w_3 \\ w_4 \\ w_5 \\ w_6 \\ w_7 \\ w_8 \\ w_9 \end{bmatrix} \in \mathbb{R}^{10}.$$

First step to performing ML parameter estimation is choosing a suitable parametric model, which we know for linear regression (still "linear" in the parameter space) is the conditional univariate Gaussian $p(y|\tilde{x};\theta) = N(y|w^T\tilde{x}, \sigma^2)$, Given a dataset D of N iid samples, this can be expressed as:

$$p(\mathcal{D} \mid \boldsymbol{\theta}) = \prod_{i=1}^{N} p(y^{(i)} \mid \tilde{\mathbf{x}}^{(i)}; \boldsymbol{\theta}) = \prod_{i=1}^{N} \mathcal{N}(y^{(i)} \mid \mathbf{w}^\intercal \tilde{\mathbf{x}}^{(i)}, \sigma^2)$$

Where $\theta = [w, \sigma^2]$,To obtain the $\check{\theta}_{MLE}$ ML estimator for the parameters of our dataset, we maximize the log-likelihood of the data. Alternatively, we can minimize the negative log-likelihood (NLL).

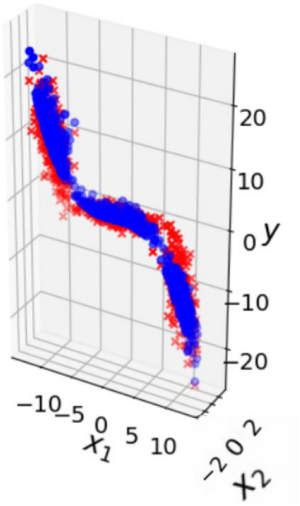
$$\begin{aligned}
\hat{\boldsymbol{\theta}}_{\text{MLE}} = \underset{\boldsymbol{\theta}}{\operatorname{argmin}}\, \text{NLL}(\boldsymbol{\theta}) &= \underset{\boldsymbol{\theta}}{\operatorname{argmin}} - \ln p(\mathcal{D} \mid \boldsymbol{\theta}) \\
&= \underset{\boldsymbol{\theta}}{\operatorname{argmin}} - \sum_{i=1}^{N} \ln p(y^{(i)} \mid \tilde{\mathbf{x}}^{(i)}; \boldsymbol{\theta}) \\
&= \underset{\boldsymbol{\theta}}{\operatorname{argmin}} - \sum_{i=1}^{N} \ln \left[\left(\frac{1}{2\pi\sigma^2} \right)^{\frac{1}{2}} \exp\left(-\frac{1}{2\sigma^2} (y^{(i)} - \mathbf{w}^\intercal \tilde{\mathbf{x}}^{(i)})^2 \right) \right] \\
&= \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \frac{N}{2} \ln(2\pi\sigma^2) + \frac{1}{2\sigma^2} \sum_{i=1}^{N} (y^{(i)} - \mathbf{w}^\intercal \tilde{\mathbf{x}}^{(i)})^2.
\end{aligned}$$

If we assume a fixed variance, we can concentrate on estimating the regression weights (w) only. This leads to an expression equivalent to the residual sum of squares (RSS) loss. In the case of negative log-likelihood (NLL) where w is the only parameter, the derivative is:

$$\nabla \text{NLL}_{\mathbf{w}}(\boldsymbol{\theta}) = \sum_{i=1}^{N} (y^{(i)} - \mathbf{w}^\intercal \tilde{\mathbf{x}}^{(i)}) \tilde{\mathbf{x}}^{(i)} = \mathbf{X}^\intercal \mathbf{y} - \mathbf{X}^\intercal \mathbf{X} \mathbf{w},$$

By representing the input features as a design matrix (x) containing the transformed input features, we can obtain the final ML estimator. To solve for this estimator, we set the gradient to zero and rearrange the equation (assuming X^TX is invertible) to gain:

$$\hat{\boldsymbol{\theta}}_{\text{MLE}} = (\mathbf{X}^\intercal \mathbf{X})^{-1} \mathbf{X}^\intercal \mathbf{y} = \left(\sum_{i=1}^{N} \tilde{\mathbf{x}}^{(i)} \tilde{\mathbf{x}}^{(i)\intercal} \right)^{-1} \left(\sum_{i=1}^{N} \tilde{\mathbf{x}}^{(i)} y^{(i)} \right) = \mathbf{w}^*.$$



Plot 2.3 ML Estimation

MAP Estimation:

The MAP Estimation equation can be concluded as below:

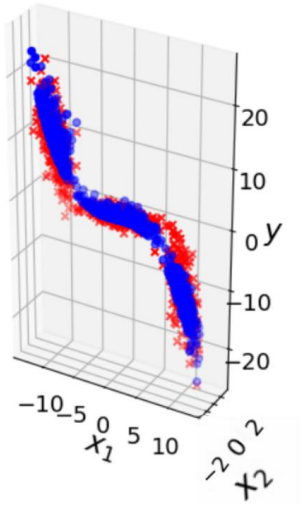
$$\hat{\theta}_{map} = \left\{ \frac{1}{N} \sum_{i=1}^{N} (y_i - f(x_i, \theta)^T \sum (y_i - f(x_i, \theta) + \frac{1}{N} (\theta - m)^T C^{-1} (\theta - m) \right\}$$

$$= \frac{1}{N} \sum_{i=1}^{N} \left\| y_i - Z_i^T \theta \right\|^2 + \frac{1}{N} (\theta - m)^T C^{-1} (\theta - m)$$

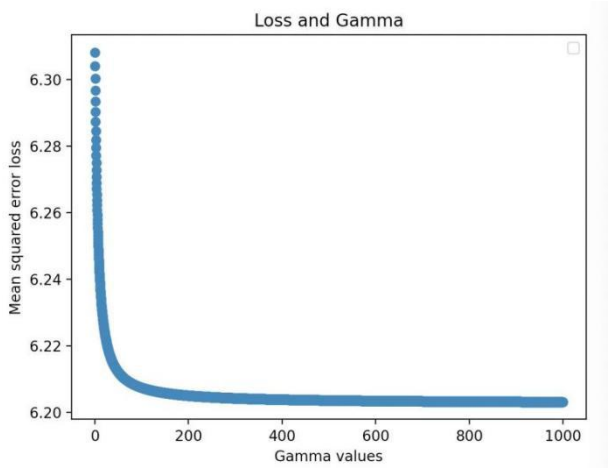
Where $\bar{z} = \frac{1}{N} \sum_{i=1}^{N} (Z_i Z_i^T)$, $\bar{y} = \frac{1}{N} \sum_{i=1}^{N} (y_i)$, after setting the equation to zero and rearranging, we can obtain our optimal MAP estimate:

$$0 = \frac{-2}{N} \sum_{i=1}^{N} (y_i - Z_i Z_i^T \theta) + \frac{2}{N} C^{-1} (\theta - m)$$

$$\hat{\theta}_{map} = \left(\bar{Z} + \frac{1}{N} C^{-1} \right)^{-1} (\bar{y} - \frac{1}{N} C^{-1} m)$$



Plot 2.4 MAP Estimation



When the value of gamma is increased, the MAP estimator begins to approach the result of the ML estimator. This happens because gamma is inversely proportional to the prior's variance, and as gamma increases, the prior's variance decreases. This leads to the posterior distribution becoming narrower around the ML point estimate, causing the posterior to be solely determined by the likelihood function, and thus MAP becomes equivalent to ML.

However, if gamma is too large or too small, the optimization objective is overly focused on maximizing the likelihood of the data, which can lead to overfitting or underfitting, respectively. This is because when gamma is too small, the posterior distribution remains too close to the prior and does not account for the available data, causing underfitting. On the other hand, when gamma is too large, the optimization objective becomes too focused on the data likelihood, which can cause overfitting.

A plot of MSE versus gamma revealed a gradual decrease in the MSE loss over the gamma interval when varying the value of gamma between 1e-5 and 1e2.

Question 3

The optimization problem can be written as:

$$\begin{aligned}\begin{bmatrix} x_{MAP} \\ y_{MAP} \end{bmatrix} &= \underset{\begin{bmatrix} x \\ y \end{bmatrix}}{argmax} \ln p(\begin{bmatrix} x \\ y \end{bmatrix}, r_1, r_2, \ldots, r_K) \\ &= \underset{\begin{bmatrix} x \\ y \end{bmatrix}}{argmax} \ln p(\begin{bmatrix} x \\ y \end{bmatrix}) p(r_1, r_2, \ldots r_K | \begin{bmatrix} x \\ y \end{bmatrix}) \\ &= \underset{\begin{bmatrix} x \\ y \end{bmatrix}}{argmax} (\ln p(\begin{bmatrix} x \\ y \end{bmatrix}) + \sum_{i=1}^{K} \ln p(r_i | \begin{bmatrix} x \\ y \end{bmatrix}))\end{aligned}$$

Based on the assumption of prior knowledge of the vehicle position, we can state that:

$$\begin{aligned}\ln p(\begin{bmatrix} x \\ y \end{bmatrix}) &= (2\pi\sigma_x\sigma_y)^{-1} e^{-\frac{1}{2}[x \quad y]\begin{bmatrix} \sigma_x^2 & 0 \\ 0 & \sigma_y^2 \end{bmatrix}^{-1}\begin{bmatrix} x \\ y \end{bmatrix}} \\ &= -\ln(2\pi\sigma_x\sigma_y) - \frac{1}{2}[x \quad y]\begin{bmatrix} \sigma_x^2 & 0 \\ 0 & \sigma_y^2 \end{bmatrix}^{-1}\begin{bmatrix} x \\ y \end{bmatrix}\end{aligned}$$

Because $-\ln(2\pi\sigma_x\sigma_y)$ is a constant, it's no contribute to argmax, so it can be removed,then according to the r_i:

$$\sum_{i=1}^{K} \ln p(r_i | \begin{bmatrix} x \\ y \end{bmatrix})) = \sum_{i=1}^{K} \ln p(d_i + n_i | \begin{bmatrix} x \\ y \end{bmatrix}))$$

Since n_i is a zero mean Gaussian measurement, we can say that the distance $d_i=n_i-r_i$, so we can

simplify the optimization problem as below:

$$\begin{aligned}
\begin{bmatrix} x_{MAP} \\ y_{MAP} \end{bmatrix} &= \underset{\begin{bmatrix} x \\ y \end{bmatrix}}{argmax}\left(-\frac{1}{2}\begin{bmatrix} x & y \end{bmatrix}\begin{bmatrix} \sigma_x^2 & 0 \\ 0 & \sigma_y^2 \end{bmatrix}^{-1}\begin{bmatrix} x \\ y \end{bmatrix} + \sum_{i=1}^{K} \ln \frac{1}{\sqrt{2\pi\sigma_i^2}} e^{-\frac{(d_i - r_i)^2}{2\sigma_i^2}}\right) \\
&= \underset{\begin{bmatrix} x \\ y \end{bmatrix}}{argmax}\left(-\frac{1}{2}\begin{bmatrix} x & y \end{bmatrix}\begin{bmatrix} \sigma_x^2 & 0 \\ 0 & \sigma_y^2 \end{bmatrix}^{-1}\begin{bmatrix} x \\ y \end{bmatrix} - \sum_{i=1}^{K} \frac{(d_i - r_i)^2}{2\sigma_i^2}\right) \\
&= \underset{\begin{bmatrix} x \\ y \end{bmatrix}}{argmax}\left(-\begin{bmatrix} x & y \end{bmatrix}\begin{bmatrix} 1/\sigma_x^2 & 0 \\ 0 & 1/\sigma_y^2 \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} - \sum_{i=1}^{K} \frac{(d_i - r_i)^2}{\sigma_i^2}\right) \\
&= \underset{\begin{bmatrix} x \\ y \end{bmatrix}}{argmin}\left(\frac{x^2}{\sigma_x^2} + \frac{y^2}{\sigma_y^2} + \sum_{i=1}^{K} \frac{(d_i - r_i)^2}{2\sigma_i^2}\right)
\end{aligned}$$

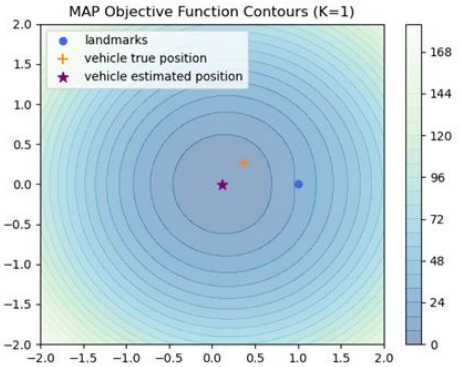
Thus,

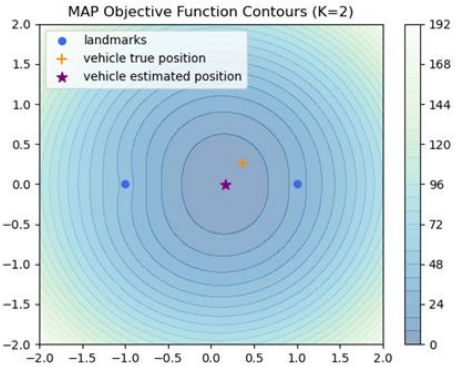
$$\begin{bmatrix} x_{MAP} \\ y_{MAP} \end{bmatrix} = \underset{\begin{bmatrix} x \\ y \end{bmatrix}}{argmin}\left(\frac{x^2}{\sigma_x^2} + \frac{y^2}{\sigma_y^2} + \sum_{i=1}^{K} \frac{(d_i - r_i)^2}{2\sigma_i^2}\right)$$

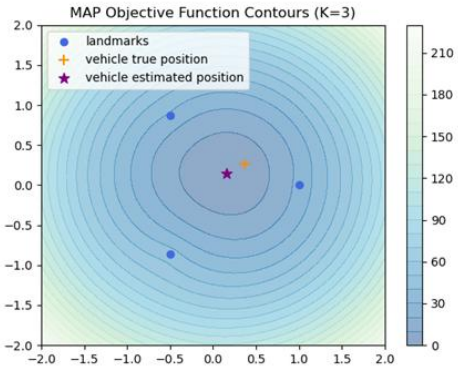
Set$\sigma_x = 0.25, \sigma_y = 0.25$,and the noise standard deviation$\sigma_1 = \sigma_2, = \ldots = \sigma_k = 0.3$

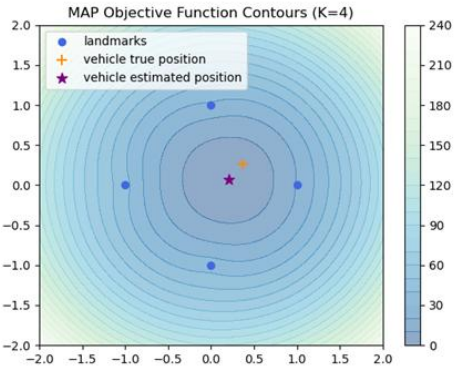
To implement the process, we begin by generating landmarks on a circle with a unit radius centered at the origin. We then determine the true position of the vehicle within the circle. Afterward, we randomly generate range measurements according to the Gaussian model mentioned above for estimation purposes.

Using our expression for the optimization problem, we can plot the contours of the MAP objective function for each value of 1 to 4 as follows:









The plot indicates that the estimated position of the vehicle is more likely to be inside the circle used for initialization. As we move outside this circle, the probability of the vehicle's location decreases rapidly (as evidenced by the denser contour lines).
To clarify the relationship between the actual position of the vehicle and the estimated position, we can calculate the distance between them. This yields:

K=1	K=2	K=3	K=4
0.3697	0.3353	0.2455	0.2561

Based on the results obtained, we can hypothesize that as the number of landmarks increases, the distance between the actual position of the vehicle and the estimated position tends to decrease, although in this instance, d(k=4)<d(k=3). This could be due to the fact that the maximum value of k is too small, resulting in a less generalized experiment. Nonetheless, we believe that as the number of landmarks increases, the MAP estimate will get closer to the true position, since a larger number of samples in the training dataset will lead to a more accurate model.

Question 4

We have a classification problem with a choice for rejection:

$$\lambda(\alpha_i|\omega_j) = \begin{cases} 0, i = j, i, j = 1, \ldots, c \\ \lambda_r, i = c + 1 \\ \lambda_s, otherwise \end{cases}$$

The risk of acting α_i is:

$$R(\alpha_i|\omega_j) = \sum_{j=1}^{c} \lambda(\alpha_i|\omega_j) p(\omega_j|\mathrm{x})$$

$$= \sum_{j=1, j\neq i}^{c} \lambda_s p(\omega_j|\mathrm{x})$$

$$= \lambda_s \sum_{j=1, j\neq i}^{c} p(\omega_j|\mathrm{x})$$

$$= \lambda_s (1 - p(\omega_j|\mathrm{x}))$$

For taking the rejection decision we have:

$$R(\alpha_{c+1}|\mathrm{x}) = \sum_{j=1}^{c} \lambda(\alpha_{c+1}|\omega_j)p(\omega_j|\mathrm{x})$$

$$= \sum_{j=1}^{c} \lambda_{\mathrm{r}} p(\omega_j|\mathrm{x})$$

$$= \lambda_{\mathrm{r}} \sum_{j=1}^{c} p(\omega_j|\mathrm{x})$$

$$= \lambda_{\mathrm{r}}$$

Action α_i is taken if its risk is smaller than the risk of taking another action α_j, $j \neq i$:

$$R(\alpha_i|\mathrm{x}) <= R(\alpha_j|\mathrm{x}), \forall j = 1, \ldots, c, j \neq i$$
$$\lambda_{\mathrm{s}}(1 - p(\omega_i|\mathrm{x})) <= \lambda_{\mathrm{s}}(1 - p(\omega_j|\mathrm{x})), \forall j = 1, \ldots, c, j \neq i$$
$$p(\omega_i|\mathrm{x}) >= (\omega_j|\mathrm{x}), \forall j = 1, \ldots, c, j \neq i$$

Also, the risk of action α_i must be less than the risk of rejection.

$$R(\alpha_i|\mathrm{x}) <= R(\alpha_{c+1}|\mathrm{x})$$
$$\lambda_{\mathrm{s}}(1 - p(\omega_i|\mathrm{x})) <= \lambda_{\mathrm{r}}$$
$$p(\omega_i|\mathrm{x}) >= 1 - \lambda_{\mathrm{r}}/\lambda_{\mathrm{s}}$$

Then, if $\lambda_r < \lambda_s(1 - P(\omega_i|\mathbf{x}))$, which is $P(\omega_i|\mathbf{x}) < 1 - \frac{\lambda_r}{\lambda_s}$, we reject it as being unrecognizable anyway, otherwise if $P(\omega_i|\mathbf{x}) \geq 1 - \frac{\lambda_r}{\lambda_s}$, we still decide ω_i as stated above.

Therefore, the statement that the minimum risk is obtained when we decide ω_i if $P(\omega_i|\mathbf{x}) \geq P(\omega_j|\mathbf{x})$ for all j and if $P(\omega_i|\mathbf{x}) \geq 1 - \frac{\lambda_r}{\lambda_s}$, and reject otherwise has been proved.

If $\lambda_r = 0$, we have:

1. If $P(\omega_i|\mathbf{x}) = 1$, both rejection and determination on ω_i are acceptable. In this situation, rejection costs nothing, while the classifier is 100% sure that $\mathbf{x}$ should be classified to class i.
2. If $P(\omega_i|\mathbf{x}) < 1$, $P(\omega_i|\mathbf{x}) < 1 - \frac{\lambda_r}{\lambda_s} = 1$ is always satisfied. According to our discussion above, we reject it anyway.

If $\lambda_r > \lambda_s$, we have $1 - \frac{\lambda_r}{\lambda_s} < 0$. Therefore $P(\omega_i|\mathbf{x}) < 1 - \frac{\lambda_r}{\lambda_s} < 0$ will never be satisfied: we will never reject any $\mathbf{x}$, and we will classify it into a class only based on the posterior probability. Here we decide ω_i if $P(\omega_i|\mathbf{x}) \geq P(\omega_j|\mathbf{x})$ for all j.

.

Question 5

MLE:

$Z=[Z_1, ..., Z_k]^T$ K-dimensional binary vector can represent the value or state of Z through a 1 to K scheme, where each element Z_K is set to 1 if the variable is in state k and 0 otherwise.

The parameter vector for the categorical distribution can be represented by $\theta=[\theta_{1,\ldots,}\theta_K]^T$,where k is

belong to 1 to K.

To obtain the maximum likelihood estimator for parameter θ, we maximize the likelihood function, which is computed based on iid samples D = {Z_1, ..., Z_N} drawn from a categorical distribution with parameter θ.

$$L(\Theta|D) = P(D|\Theta) = \prod_{k=1}^{K} \theta_k^{\delta kZn}$$

Thus,

$$\log(L(\Theta|D)) = P(D|\Theta) = \sum_{k=1}^{K} \delta_{kz} log(\theta_k)$$

Then maximizing the likelihood function with respect to θ , and use lagrange multiplier

$f(\theta,\lambda) = \log L(D|\theta)+\lambda(1-\sum_{k=1}^{K} log(\theta_k))$, where $\partial f/\partial\theta=0$, $\partial f/\partial\lambda=0$.

As a result,

$\theta_k^{ML} = N_k/N, for\ k \in \{1,\dots,K\}.$, where $N_k=\sum_{n=1}^{N} \delta Zk$

MAP:

Assuming that the prior p(Θ) for the parameters is a Dirichlet distribution with hyperparameter α,

Assuming that the prior p(θ) for the parameters is a Dirichlet distribution with hyperparameter α,

the posterior distribution for Θ given the dataset D is:

$$p(\Theta|D) = \frac{1}{B(\alpha)} \prod_{k=1}^{K} \theta_k^{(\sum_{n=1}^{N} \delta Zk+\alpha_k-1)} \prod_{k=1}^{K} \theta_k^{\delta Zk}$$

Thus,

$$\log(p(\Theta|D)) = \sum_{k=1}^{K} \delta_{zk}\log(\theta_k)+\sum_{k=1}^{K} \alpha_{k-1}\log(\theta_k)$$

Taking derivative , as a result:

$\theta_k^{MAP} = \frac{\sum_{n=1}^{N} Z_n k+\alpha_{k-1}}{\sum_{k=1}^{K}\sum_{n=1}^{N} Z_n k+\alpha_{k-k}} = \frac{N_k+\alpha_{k-1}}{N+\sum_{k=1}^{K}\alpha_{k-k}}, for\ k \in \{1,\dots,K\}.$, where $N_k=\sum_{n=1}^{N} \delta Zk$

Append

The code of project2 is as follow link:

https://github.com/JonnyFan/ML5644/tree/main/hw2