Anush Sriram Ramesh

Machine Learning and Pattern Recognition EECE 5644 V30

HW 4

[GitHub Link](https://github.com/sriram-0311/MLPR.git) **– Go into folder HW4 for code and results**

**Question 1.**

**1.PART (1) Theoretically Optimal Classifier for Minimum P of Error**

From HW2, the theoretically optimal classifier for the dataset is likelihood ratio test.

Equation 1

Where, λ is the cost matrix associated with the decisions.

Let us assume [0 – 1] cost matrix for minimizing the probability of error.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **ACTUAL TRUTH** | | |
| **DECISIONS** | **LABEL** | 0 | 1 |
| 0 | 0 | 1 |
| 1 | 1 | 0 |

Substituting values of λ back into the equation 1,

Data extracted from the question,

,

where, w1 = w2 = 0.5 and means, covariances parameters as per the question.

Chart, scatter chart

Description automatically generated

**Fig: Training samples generated as per the likelihood function. Class 1 is P(X|L=0) and Class 2 is P(X|L=1)**

**ROC Curve for Theoretically optimal classifier:**

From the above equation, Probability of error and threshold corresponding to the probability can be calculated. It can also be found empirically by varying the threshold in the code.

Table 1

|  |  |  |
| --- | --- | --- |
|  | **THEORETICAL** | **EMPERICAL** |
| **PROBABILITY OF ERROR** | 0.175 | 0.16328 |
| **THRESHOLD** | 1.5 | 1.41 |

Chart, scatter chart

Description automatically generatedChart

Description automatically generated

**Fig: ROC Curve generated for likelihood ratio classifier with theoretical and empirical thresholds for minimum P of error marked. OPTIONAL SOLUTION Best classifier decision boundary achieved by the likelihood ratio test.**

The AUC for the curve is high (close to 1) indicating that the classifier performs well on the data.

**OPTIONAL:** It is apparent from the contour of the decision boundary that it indeed is a very good classifier for the given data.

We’ve to try to implement an estimated classifier which performs close to this classifier.

**1.PART (2) Parameter estimation for Gaussian Mixture Model using EM Algorithm**

We ignore the parameters mentioned in the question for this part. We will try to estimate the class priors, mean and covariance for class labels 0 and 1.

**Estimating Parameters for Class 0**

Estimating parameters for class 0 involved estimating the weights of the gaussian (w1 , w2) and the corresponding means and covariances (m01, c01) and (m02, c02).

EM algorithm estimates the parameters of the GMM by maximizing the likelihood function given the prior estimation Θg (parameters of GMM) .

Here, is the prior estimate of the model parameters. vector contains (w1 , w2, …. wm, m01, c01 , m02, c02 ….. mm, cm) – Formal definition of EM estimator.

Updates used by the estimator are

For Class 0,

For Class 1, there is only one gaussian. Hence the mean and covariance are the sample mean and covariance.

Class priors are estimated simply as the percentage of samples in Class label 0 and label 1 as we know the labels for samples in training data.

**Results for Training MLE with 10K Samples**

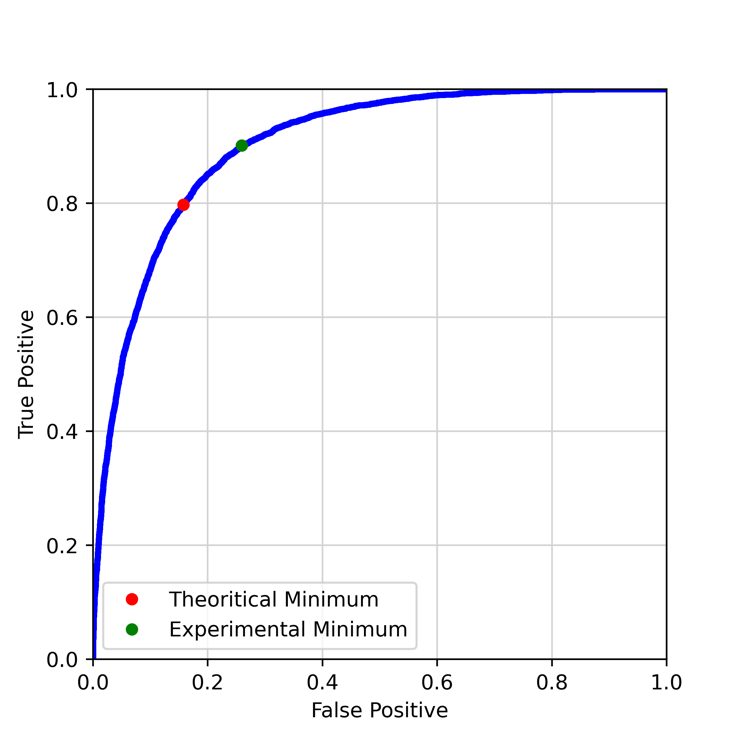
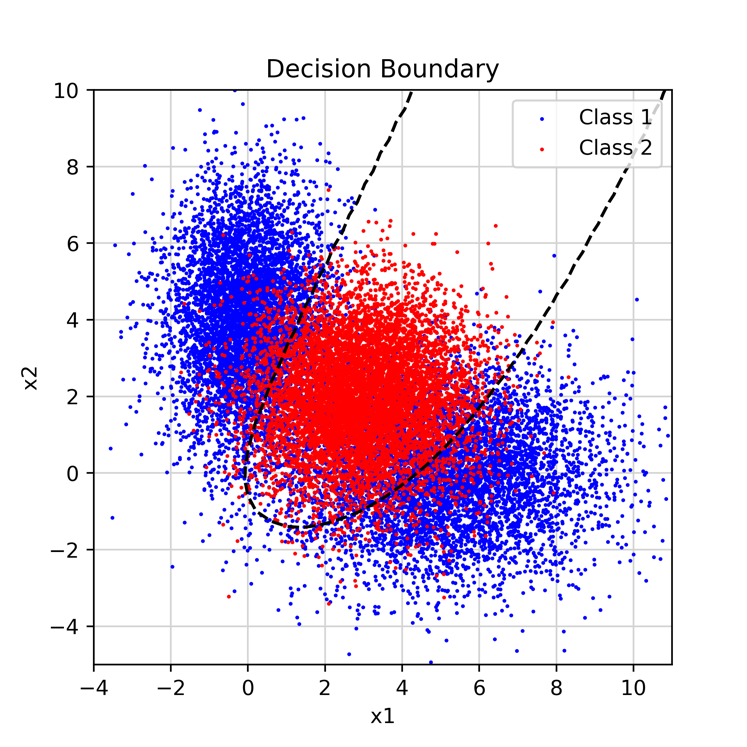
Using the **GMM** in **SKLearn.mixture** in python to fit a gaussian mixture model to the samples. Below figures show the contours plotted with the estimated gaussian parameters received from that function.

**Class Label 0**

**Class Label 1**

Chart, diagram

Description automatically generated**Fig: Contour plots of the estimated gaussian distributions for each class label generated by EM algorithm.**

Using the estimated gaussian parameters run the Likelihood ratio classifier on the data and report the ROC Curve and Probability of error.

**Fig: ROC Curve for Likelihood ratio test with estimated Gaussian parameters trained with 10K samples and the contour of the decision boundary.**

The minimum probability of error achievable for this estimation and the threshold found empirically for that.

Table 2

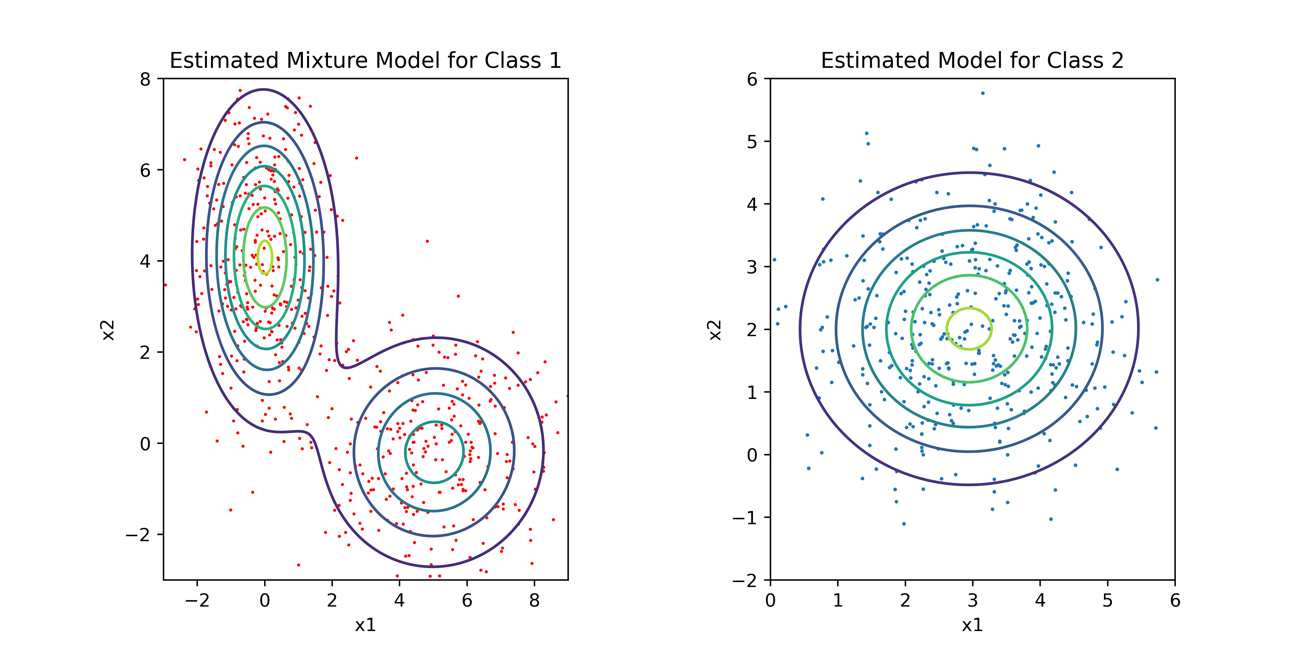
|  |  |  |
| --- | --- | --- |
|  | **THEORETICAL** | **EMPERICAL** |
| **PROBABILITY OF ERROR** | 0.175 | 0.163 |
| **THRESHOLD** | 1.481 | 1.4015 |

The estimator has estimated the gaussian very close to the actual parameters. Hence, the classifier performs well with estimated parameters too.

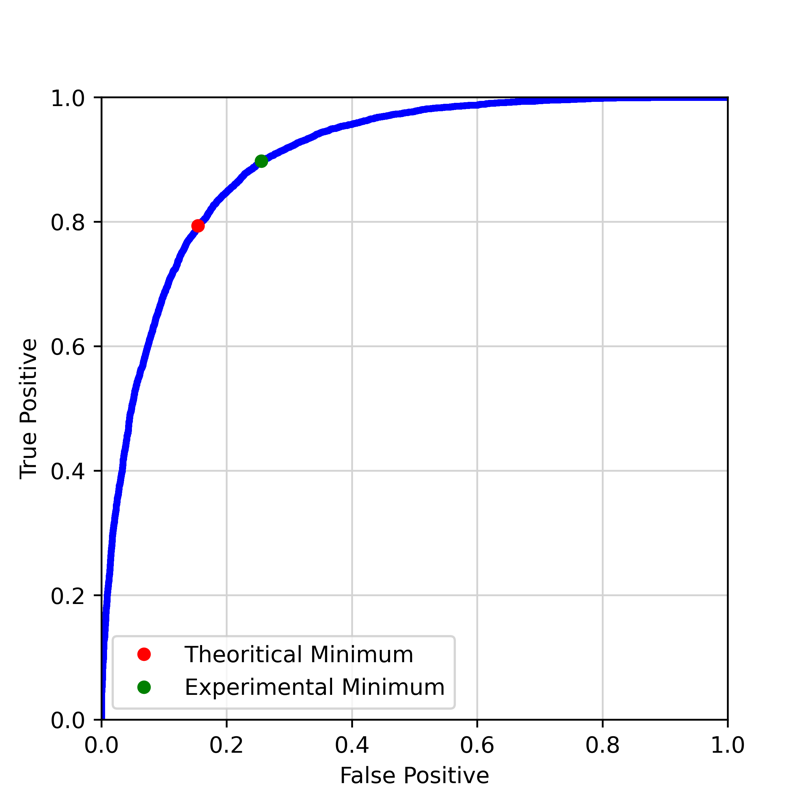
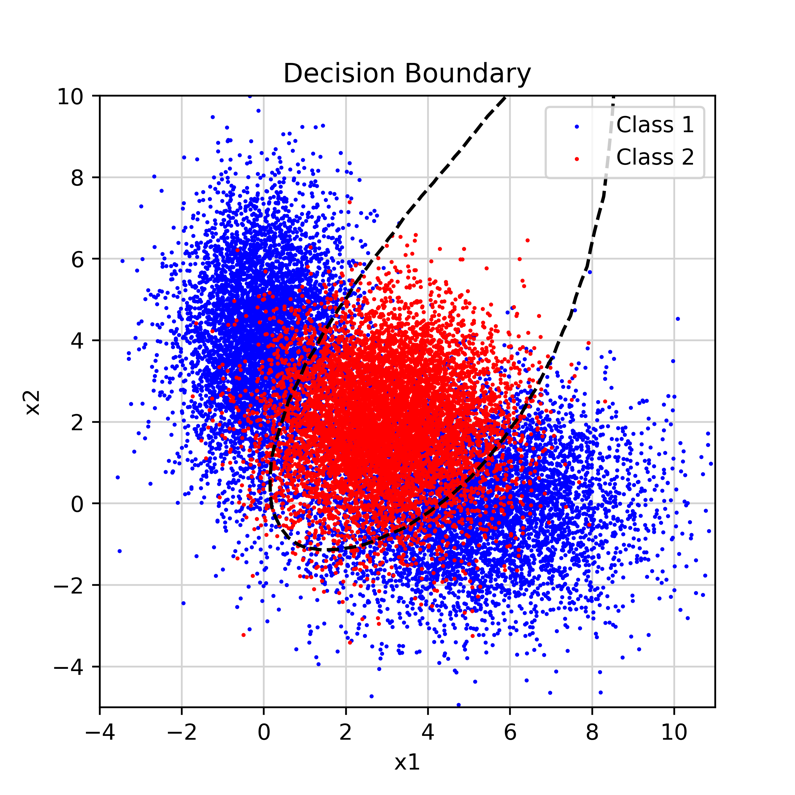
**Results for Training MLE with 1000 Samples**

**Class Label 0**

**Class Label 1**



**Fig: Contour plots of the estimated gaussian distributions for each class label generated by EM algorithm.**

Using the estimated gaussian parameters run the Likelihood ratio classifier on the data and report the ROC Curve and Probability of error.

**Fig: ROC Curve for Likelihood ratio test with estimated Gaussian parameters trained with 1000 samples and the contour of the decision boundary.**

The minimum probability of error achievable for this estimation and the threshold found empirically for that.

Table 3

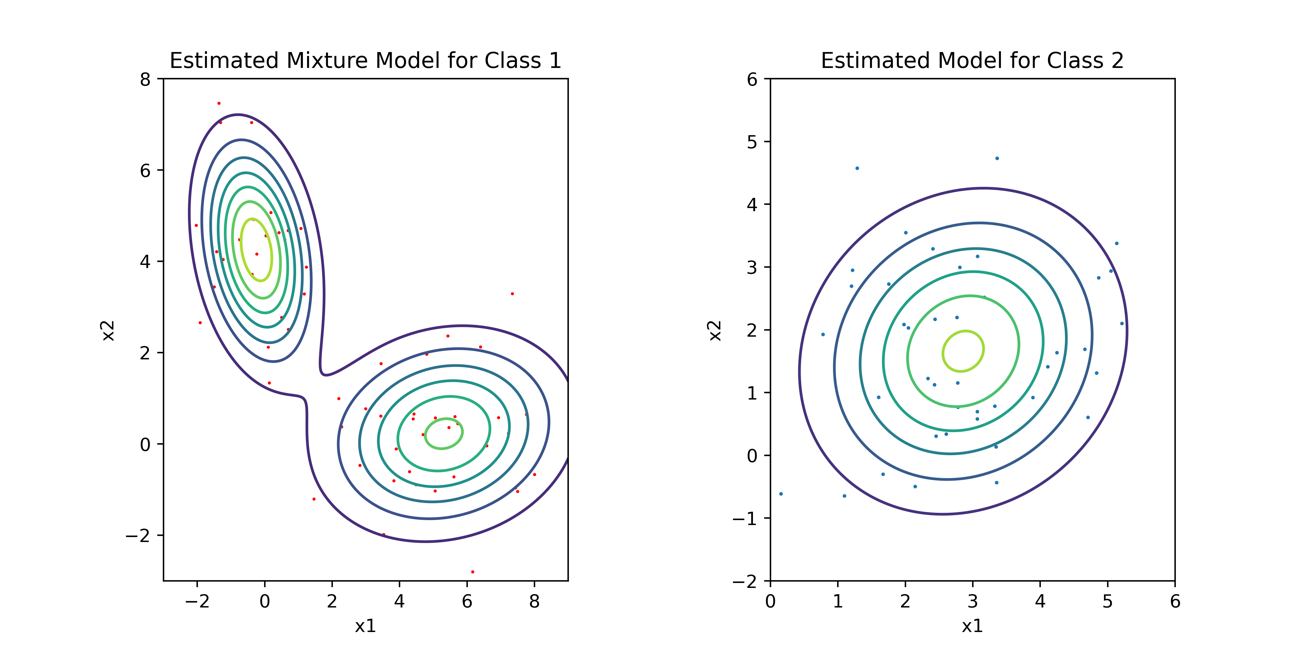
|  |  |  |
| --- | --- | --- |
|  | **THEORETICAL** | **EMPERICAL** |
| **PROBABILITY OF ERROR** | 0.174 | 0.162 |
| **THRESHOLD** | 1.525 | 1.498 |

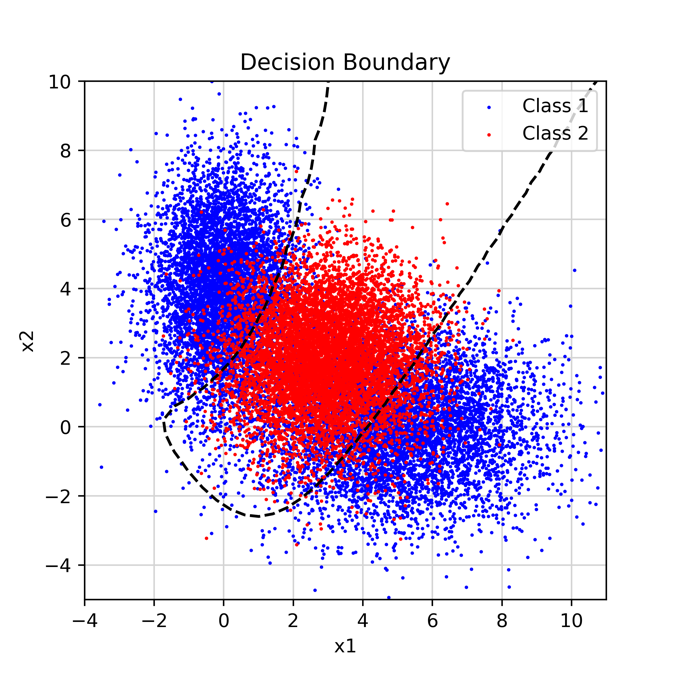
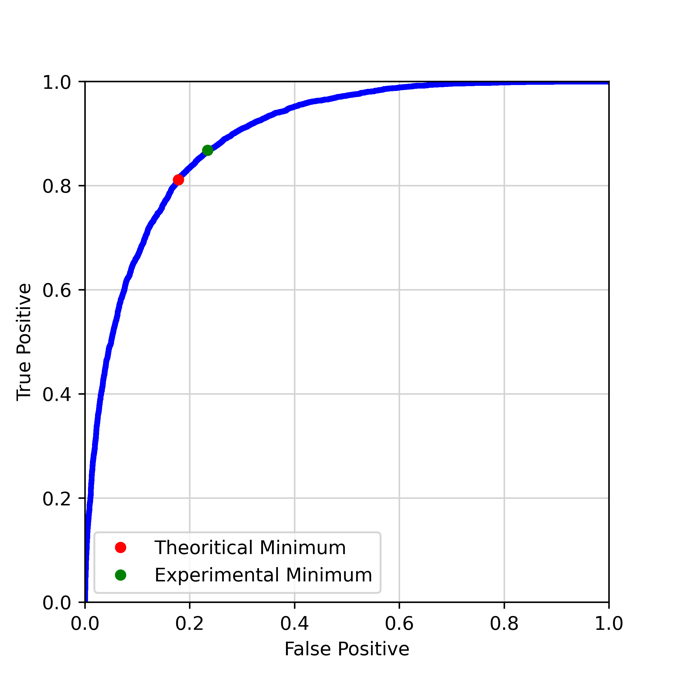
The estimator wasn’t as good as with 10K samples, but still performed a decent job of estimating the parameters, which in most cases might just be optimal performance to number of samples ratio.

**Results for Training MLE with 100 Samples**

**Class Label 0**

**Class Label 1**

**Fig: Contour plots of the estimated gaussian distributions for each class label generated by EM algorithm.**

Using the estimated gaussian parameters run the Likelihood ratio classifier on the data and report the ROC Curve and Probability of error.

**Fig: ROC Curve for Likelihood ratio test with estimated Gaussian parameters trained with 100 samples and the contour of the decision boundary.**

The minimum probability of error achievable for this estimation and the threshold found empirically for that.

Table

|  |  |  |
| --- | --- | --- |
|  | **THEORETICAL** | **EMPERICAL** |
| **PROBABILITY OF ERROR** | 0.1825 | 0.189 |
| **THRESHOLD** | 1.5439 | 1.443 |

As expected, the estimator produced bad estimates for the gaussian parameters with just 100 samples. Thus, the likelihood ratio test also performs worser than with 1000 samples and with 10K samples, although it isn’t that terrible with such low samples.

The Minimum P-Error doesn’t increase significantly when the number of samples is reduced from 10K to 1000, it does increase when the number of samples is furthered decreased to 100.

Which shows, 10K samples is too much even though it gives a pretty accurate estimate of the gaussians, 1000 samples achieve almost similar performance with lower samples. 100 samples though is too low number of samples to achieve a reliable estimate of the gaussians.

**1.PART (3) Logistic Linear Regression and Logistic Quadratic Regression**

**LLR Classifier**

Now we train LLF classifier to approximate a linear decision boundary that transforms each input vector x via b into (1, x1,x2).

LLR is a linear boundary, hence the performance of LLF is going to be much worser than the likelihood ratio test.

Table 4

|  |  |  |
| --- | --- | --- |
|  | Chart, scatter chart  Description automatically generated | Chart, scatter chart  Description automatically generated |
| **Fig: LLR Classifier trained with 10K samples. Min P-Error = 0.40105**  **Threshold for Min P-Error = 0.62** | **Fig: LLR Classifier trained 1000 samples**  **Min P-Error = 0.40105**  **Threshold = 0.79** | **Fig: LLR Classifier trained 100 samples**  **Min P-Error = 0.40105**  **Threshold = 0.901** |

All lines generated in LLR are bad classifiers of the data, hence no improvement in performance despite having more samples.

**LQR (OPTIONAL)**

Linear logistic regression was totally inappropriate for the data set, let’s try to fit quadratic regressor to the dataset to see if we perform better. Now the input vector is transformed via (1, x1,x2, x12,x1 \* x2,x22). Since the transforming vector has quadratic terms on both axes it is capable of producing nonlinear decision boundary.

Table 5

|  |  |  |
| --- | --- | --- |
|  |  |  |
| **Fig: LLR Classifier trained with 10K samples. Min P-Error = 0.175**  **Threshold for Min P-Error = 0.550** | **Fig: LLR Classifier trained 1000 samples**  **Min P-Error = 0.175**  **Threshold = 0.5725** | **Fig: LLR Classifier trained 100 samples**  **Min P-Error = 0.2025**  **Threshold = 0.5399** |

**2. MAP Estimation**

**Text, letter

Description automatically generated2. Part 1) Optimization Problem**

**A picture containing text, person, document

Description automatically generated**

**Text, letter

Description automatically generated**

**Text

Description automatically generated2.PART 2) CODE WALKTHROUGH**

The code was tested for situations with K = 1,2,3,4 and 40.

True Point condition r ≤ ¾ evenly spaced in the circle by angle.

Measurements taken with noise 0.3 ignoring negatives.

1. We initialize the true position in the code first according to the condition.
2. Run a loop of the entire functionality, simulating conditions K = 1,2,3,4 and 40
3. For each K, compute the reference points via linspace function in python such that they are evenly spaced somewhere inside the unit circle.
4. For all i = [1,K]
   1. Init σi = 0.3
   2. Init xi = (cosθi, sinθi)
   3. Calculate yi as || xi – truepos || + N(0, σi2)
   4. Repeat c if yi < 0.
5. Define the MAP\_Loglikelihood function as derived above.
6. Plot the contours of the function using matplotlib contour and the minimum of the MAP function as the MAP estimate of the position of the vehicle with reference points.

As the number of reference points increasing, it is evident from the output of the code that the MAP performs better and gets closer to the True Position of the robot.

Chart

Description automatically generatedChart, diagram

Description automatically generatedHence, asymptotically(K → ∞), the MAP estimate becomes our MLE estimate.

**Fig: Contour plot of the likelihood function along with reference points and true position**

Chart

Description automatically generatedChart, diagram

Description automatically generatedChart

Description automatically generated

**Fig: Contour plot of the likelihood function along with reference points and true position**

It can be seen from the plots that the MAP Estimate of the position of vehicle gets closer and closer as the number of reference vectors increase. And at K = 40, it is no more dependent on the prior and pretty accurate.

**Text, letter

Description automatically generated3. Part 1) Risk classifier**

**Text, letter

Description automatically generated**

**Text, letter

Description automatically generated**

**Text, letter

Description automatically generated**