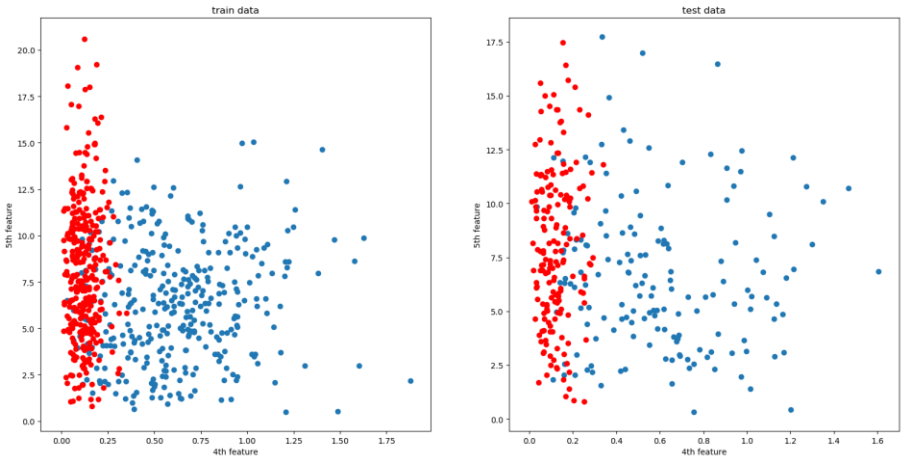

Figure 1 scatter plot of 10d dataset with 4th and 5th feature

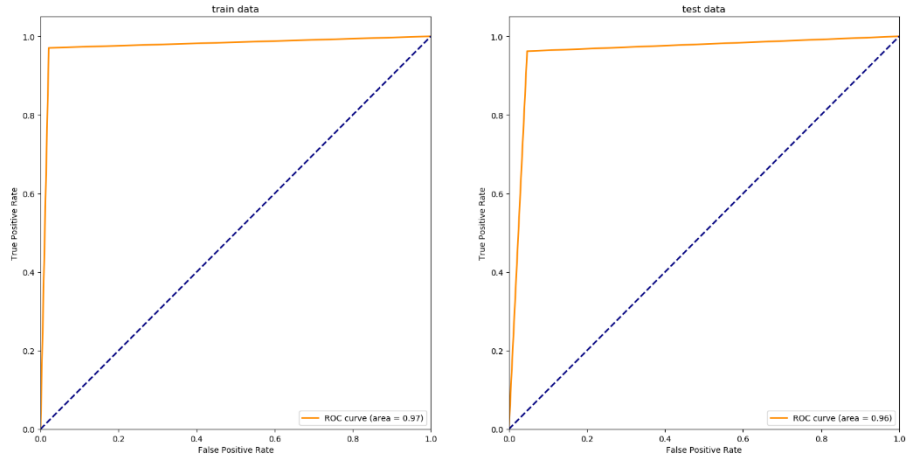

Figure 2 ROC curve of probabilistic generative classifier on 10d dataset

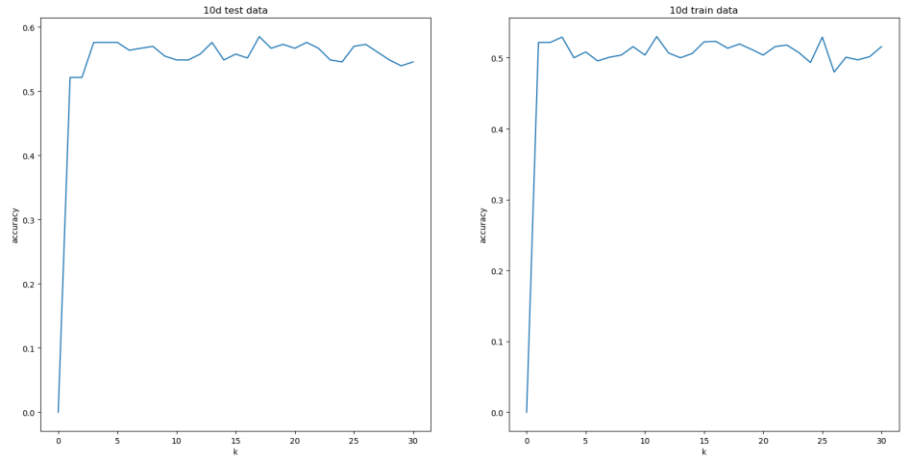

Figure 3 the accuracy as varying K for KNN

1)

Crab_Probabilistic Generative Model		
	Actual Ture	Actual False
Predicted Ture	32	0
Predicted False	0	34

10D_Probabilistic Generative Model		
	Actual Ture	Actual False
Predicted Ture	153	8
Predicted False	6	163

Crab_KNN		
	Actual Ture	Actual False
Predicted Ture	28	3
Predicted False	4	31

10D_KNN		
	Actual Ture	Actual False
Predicted Ture	96	71
Predicted False	63	100

The confusion matrixes of both KNN model are a little different after every training. Because I use cross-validation and split train data randomly.

2)

The problem is that the covariance matrix of the carb dataset is singular. My solution is regularization. I add an identity matrix of n times to the covariance matrix so that it is not singular.

3)

If the K value is small, the model complexity is high and it is easy to overfit. If K is too large, the points of different class that are far away from it become neighbors, the deviation becomes larger, and the accuracy is lower. Classification accuracy decreases as K value increases

4)

Because there are too many features in 10d dataset. And it is difficult to classify the data by Euclidean distance. To improve it, I add weight to the distance.

Also, many features in 10d dataset are very similar in both classes. So when I use KNN classifier, it is difficult to classify the data just from their neighbors. To improve it, I can find the most unique features from their 10 features and use those unique features to train the model.

5)

I will use probabilistic generative classifier for both datasets. Because after cross-validation, I find that probabilistic generative classifier has higher accuracy and stability.

Mean score about 2 model		
	Crab	10D
Probabilistic Generative	1.0	0.974
KNN	0.937	0.571

6)

1.

Data likelihood is

$$\prod_{n=1}^{N} p(x_n|\lambda) = \prod_{n=1}^{N} \frac{\lambda^{x_n}}{x_n!} e^{-\lambda}$$

Take the log:

$$\mathrm{L} = \ln \prod_{n=1}^{N} \frac{\lambda^{x_n}}{x_n!} e^{-\lambda} = \sum_{n=1}^{N} \ln\left(\frac{\lambda^{x_n}}{x_n!} e^{-\lambda}\right) = \sum_{n=1}^{N} [x_n \ln \lambda - \ln(x_n!)] - N\lambda$$

$$\frac{\partial L}{\partial \lambda} = \sum_{n=1}^{N} \frac{x_n}{\lambda} - N = 0$$

$$\rightarrow \lambda = \frac{1}{N} \sum_{n=1}^{N} x_n$$

2.

Gamma distribution:

$$p(\lambda|a) = \frac{\lambda^a e^{-\lambda}}{\Gamma(\mathrm{a})}$$

The Gamma function:

$$\Gamma(\mathrm{x}) = \int_0^{+\infty} t^{x-1} e^{-t} dt \; (x > 0)$$

If x is positive integer, $\Gamma(\mathrm{x}) = (x-1)!$

$$p(\lambda|X, a) \propto p(X|\lambda) p(\lambda|a) = \prod_{n=1}^{N} \frac{\lambda^{x_n}}{x_n!} e^{-\lambda} \cdot \frac{\lambda^a e^{-\lambda}}{\Gamma(\mathrm{a})}$$

Take the log:

$$\mathrm{L} = \ln \prod_{n=1}^{N} \frac{\lambda^{x_n}}{x_n!} e^{-\lambda} \cdot \frac{\lambda^a e^{-\lambda}}{\Gamma(\mathrm{a})} = \sum_{n=1}^{N} [x_n \ln \lambda - \ln(x_n!)] - N\lambda + a ln\lambda - \lambda - \ln\Big(\Gamma(\mathrm{a})\Big)$$

$$\frac{\partial L}{\partial \lambda} = \sum_{n=1}^{N} \frac{x_n}{\lambda} - N + \frac{a}{\lambda} - 1 = 0$$

$$\rightarrow \lambda = \frac{1}{N+1} (a + \sum_{n=1}^{N} x_n)$$