DEPARTMENT OF COMPUTER ENGINEERING AND IT

AMIRKABIR UNIVERSITY OF TECHNOLOGY

Computer Engineering && IT Department

Amirkabir University of Technology

Statistical Pattern Recognition

Submitted To:
Mohammad Rahmati
Assoc. Professor
Computer Engineering
Department

Submitted By :
Ahmad Asadi
94131091
Group-G1
Fall-95

Contents

List of Figures

List of Tables

Listings

1 Plotting conditional Parzen window based density estimates

In the Parzen window formula, variable V denotes the hypercube volume which given $h = 1$ is equal to 1. So the final formula is rewritten as (1) in which ω_i denotes the ith class.

$$\hat{p}(x|\omega_i) = \frac{1}{n}\Sigma_{j=1}^{n_i}\phi(x - x_j) \tag{1}$$

According to (1), the first class conditional density estimate of first class based on Parzen window is expressed in (2).

$$\hat{p}(x|\omega_1) = \frac{1}{5}(\phi(x-x_1)+\phi(x-x_2)+\phi(x-x_3)) = \frac{1}{5}(\phi(x-4)+\phi(x-1)+\phi(x-5)) \tag{2}$$

And that of the second class is expressed in (3).

$$\hat{p}(x|\omega_2) = \frac{1}{5}(\phi(x - x_4) + \phi(x - x_5)) = \frac{1}{5}(\phi(x - 3) + \phi(x - 2)) \tag{3}$$

Figure 1 displays both Parzen window estimated likelihood as a function of x for both classes.

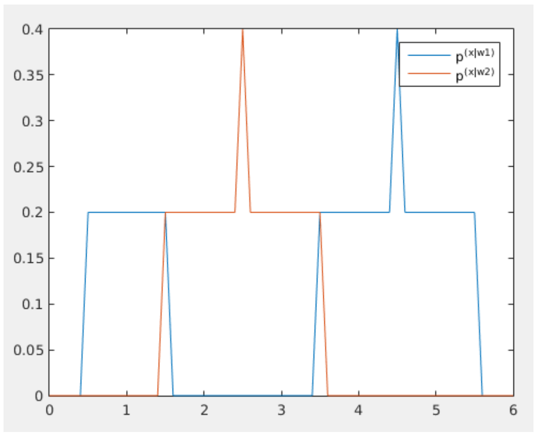


Figure 1: Estimated likelihood of class conditional densities using Parzen window.

2 Plotting Parzen window estimate using Gaussian window function

- Equation (4) represents the normal kernel.

$$\frac{1}{h^2}\phi(\frac{X-X_i}{h}) = (2\pi)^{-\frac{n}{2}}h^{-n}|\Sigma|^{\frac{1}{2}}exp[-\frac{1}{2}h^{-2}(X-X_i)^T\Sigma^{-1}(X-X_i)] \qquad (4)$$

  According to $\phi \sim N(0,1)$ the covariance matrix is $\Sigma = I$ and the mean vector is $\mu = 0$. The variable n denoting the number of dimensions is equal to 2. So the normal kernel represented in (4) is rewritten as equation (5).

$$\frac{1}{h^2}\phi(\frac{X-X_i}{h}) = (2\pi)^{-1}h^{-2}exp[-\frac{1}{2}h^{-2}(X-X_i)^T(X-X_i)] \qquad (5)$$

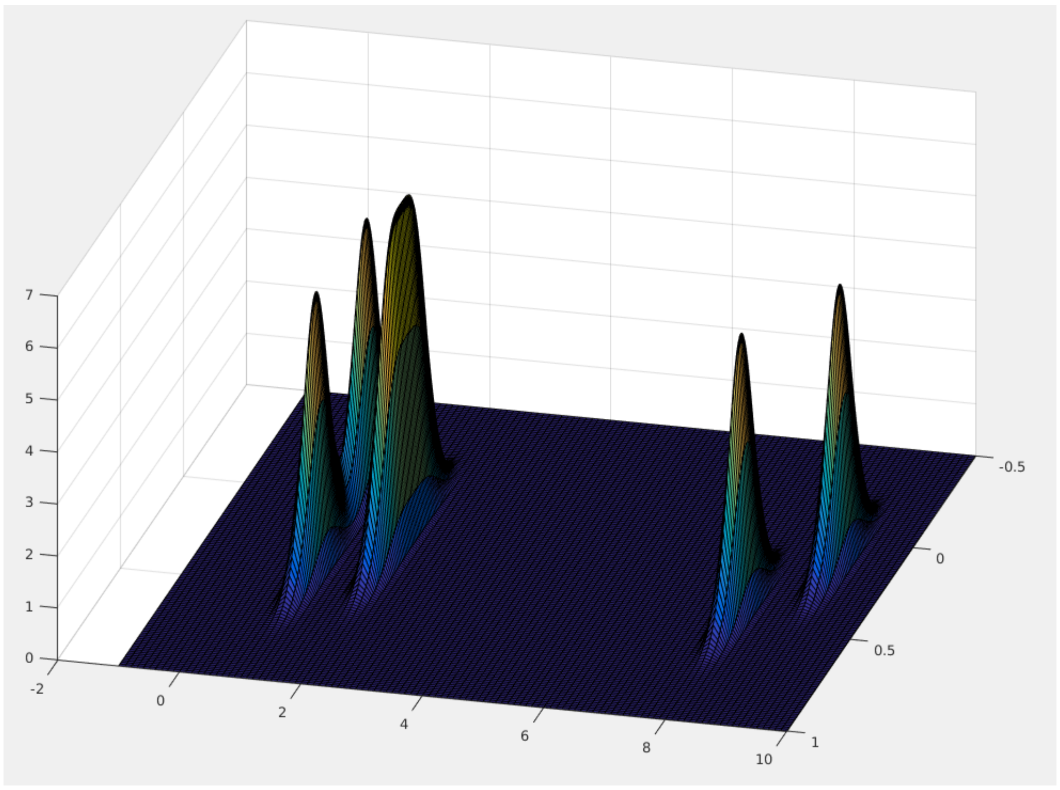
  The Parzen estimate is represented in (6) in normal kernel case.

$$\hat{P} = \frac{1}{N}\Sigma_{i=1}^{N}\frac{1}{h^2}\phi(\frac{X-X_i}{h}) = \frac{1}{2N\pi h^2}\Sigma_{i=1}^{N}exp[-\frac{1}{2}h^{-2}(X-X_i)^T(X-X_i)] \quad (6)$$

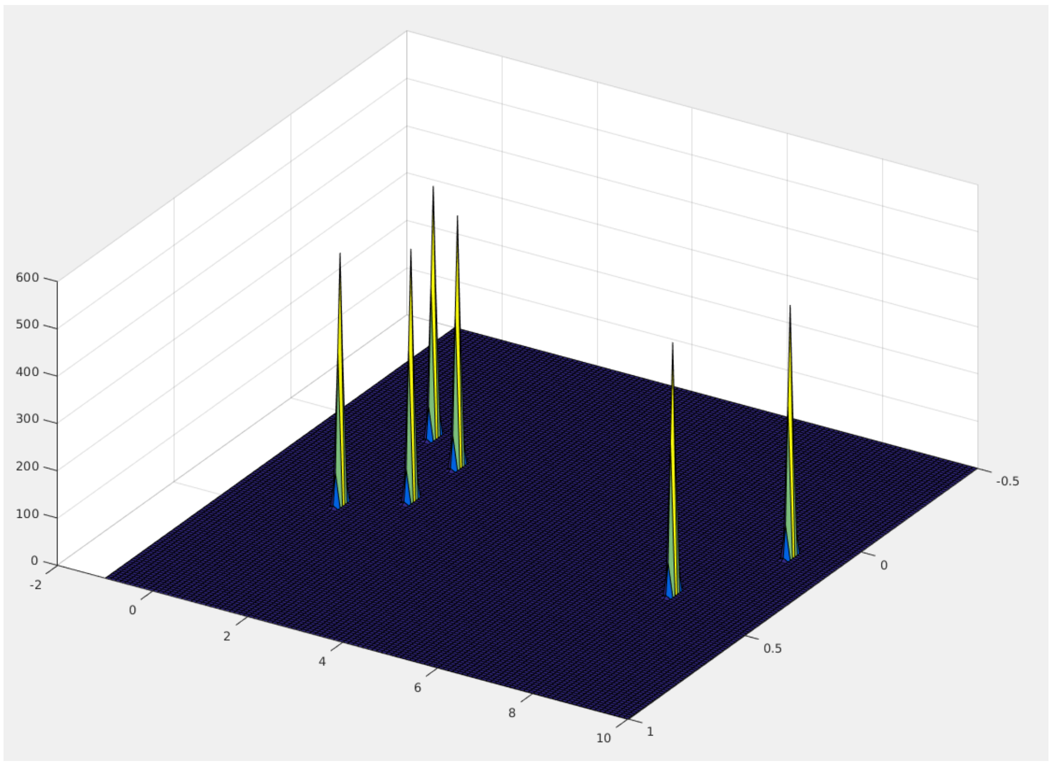
  Using given dataset $x = \{< 0.0, 1 >; < 0.1, 2 >; < 0.1, 9 >; < 0.3, 2 >; < 0.4, 1 >; < 0.4; 8 >\}$ the Parzen estimate is computable. Figure 2a displays the estimated density using $h = 0.1$ and figure 2b displays estimated density in case of $h = 0.01$.

  As it is clearly obvious, decreasing h strongly makes model finer. The estimated density function is coarser when $h = 0.1$ rather than when $h = 0.01$.

- In K-Nearest-Neighbour method, it is supposed to continuously increase averaging region centered at X until exactly K other elements locate in the region. Figure 3 displays estimated density function and its contour lines using k-Nearest-Neighbour algorithm with $k = 3$. As it is shown in the figure, in the regions containing 3 samples closer to each other, estimated density function has higher value rather than other regions. In addition, in the regions with 2 or less samples close to each other, although there exists a local extreme, the value of the extreme points is way lower than regions with 3 or more samples close to each other.

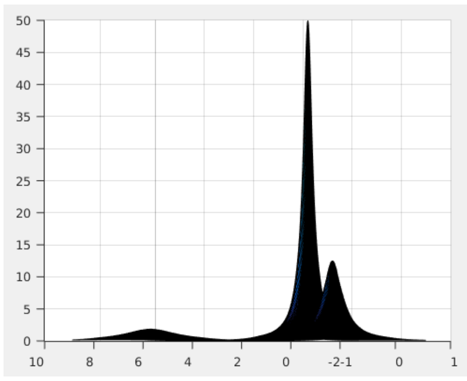


(a) Estimated density with $h = 0.1$

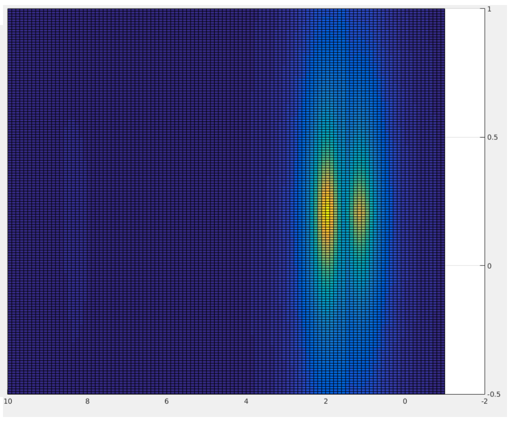


(b) Estimated density with $h = 0.01$

Figure 2: Estimated density functions using Parzen Gaussian window



(a) Estimated density function using 3NN.



(b) Estimated density function contour lines using 3NN.

Figure 3: Estimated density function and its contour lines using 3NN according to given dataset.

3 Classification Error Rate Of K-Nearest-Neighbour Algorithm

In order to compute leave-one-out cross-validation error rate of k-NN classifier, we will First classify all samples using k-NN and then calculate average on misclassified samples count. Since in the problem the exact values for real-values inputs x is not given, we have assumed that the distribution of samples is like figure 4.

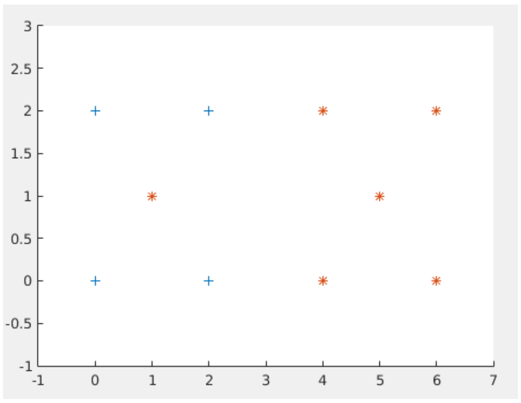
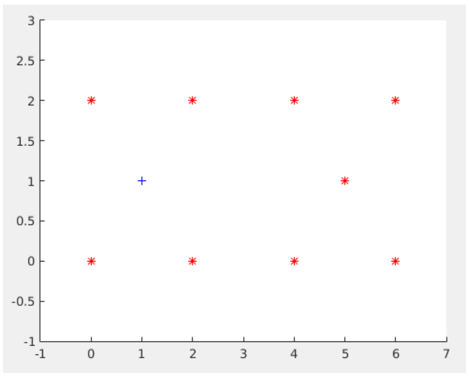


Figure 4: Assumed distribution for given dataset in problem 3.

- Classification using 1-NN will result in predictions displayed in figure 5a according to assumed distribution for training samples. As it is obvious in the figure, in this case, 4 samples of class M_+ (samples displayed by *'+'* in figure 4) and 1 sample of class M_* (samples displayed by *'*'* in figure 4) are classified incorrect. So the leave-one-out cross-validation error rate of 1-NN classifier is $\epsilon = \frac{5}{10} = \frac{1}{2}$.

- Classification using 3-NN will result in predictions displayed in figure 5b according to assumed distribution for training samples. In this case 2 samples from M_+ and 1 sample from M_* are classified incorrect which yields leave-one-out cross-validation error rate as $\epsilon = \frac{3}{10}$.


(a) Classification result using 1-NN

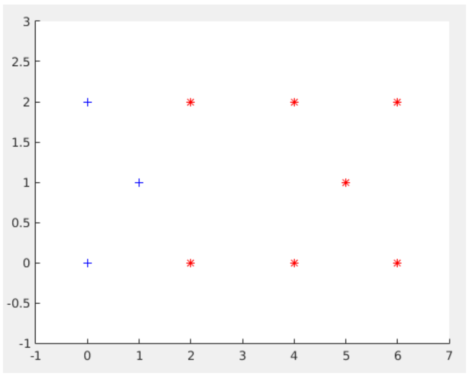

(b) Classification result using 3-NN

Figure 5: Classification results of 1-NN and 3-NN classifiers on given dataset in problem 3.

- According to a discussion in Duda et al. (2012) the error rate of K-NN algorithm can be calculated by equation (7) in which $P(\omega_i|X)$ is estimated using K-NN density function estimation.

$$P_\epsilon = \int [1 - \Sigma_{i=1}^{c} P(\omega_i|X)] P(X) dX \tag{7}$$

In order to minimize P_ϵ represented in (7) we should maximize $\Sigma_{i=1}^{c} P(\omega_i|X)$. Since there does not exist any general way to maximize $\Sigma_{i=1}^{c} P(\omega_i|X)$ for any possible dataset, the only working approach is to test different Ks to get the best one.
In addition, there is a trade-off in model's generalizability and accuracy. To avoid overfitting, it is supposed to use a penalty function on K. In other words, the larger K, the coarser model, the higher generalizability and lower accuracy on training samples. On the other hand, the lower K, the more local decisions, the higher accuracy on training samples, the more effect of outliers and the less generalizability.

Leave-one-out cross-validation error rate could be a good measure to make a reasonable trade-off and choose a suitable k. It can be done as selecting K in a reasonable range, calculating leave-one-out cross-validation error rate for each K and selecting the K which minimizes it.

4 Computer Project 1

The listing 1 represents the MATLAB script to find the leave-one-out cross-validation error rate on training samples D with labels L using K.

Listing 1: K-NN code to compute leave-one-out cross-validation error rate on training set D with labels L using K

```
1  function error = KNN_LOO(D, L, K)
2      error = 0 ;
3      for i = 1 : size(D,1)
4          X = D(i,:) ;
5          dists = (repmat(X,size(D,1),1) - D).^2 ;
6          dists(find(dists(:,3) ~= 0),3) = 1 ;
7          dists(find(dists(:,7) ~= 0),7) = 1 ;
8          dists(find(dists(:,13) ~= 0),13) = 1 ;
9          dists = sum(dists.') ;
10         [~,sortedInds] = sort(dists,'ascend') ;
11         knnLabels = L(sortedInds(2 : 2 + K),:) ;
12         l = mode(knnLabels) ;
13         if(l ~= L(i))
14             error = error + 1 ;
15         end
16     end
17     error = error / size(D,1) ;
18 end
```

Line 5 of listing 1 computes Euclidean distance between sample ith of dataset with all other points. Lines 6 to 8 are used to revise distance for nominal dimensions 3,7,13, according to data set description. Line 9 computes the distance buy summation over all dimensions. Lines 10 and 11 sorts distances in ascending order and takes top K shortest distances as K-NNs. Finally the mode of labels of K-NNs defines predicted label l for current sample. In lines 13 to 15 if the predicted label is not correct error will be increased one unit. Finally in line 17 the exact error will be calculated.
Using this function we are able to make a loop on K between 1 to 10 and find the

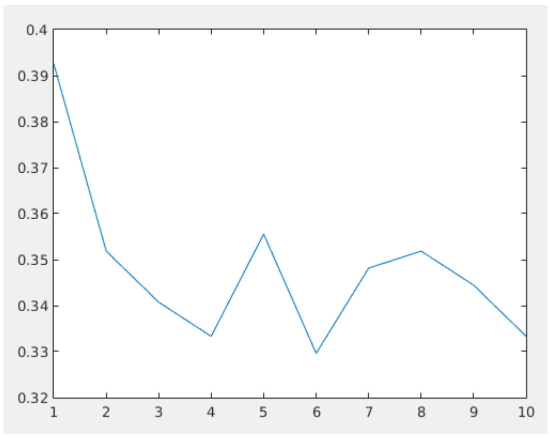


Figure 6: The leave-one-out cross-validation error rate gained for each K over given dataset in computer exercise 1.

best K with the smallest error rate.
Figure 6 represents leave-one-out cross-validation error rate for each K in the range of 1 to 10. The best error rate $\epsilon = 0.3296$ has been gained with $K = 6$ in this experiment. Also, table 1 represents the averaged error rate of the algorithm for each K w.r.t given dataset.

Table 1: Reported error rates for each K over given dataset in computer exercise 1.

Error Rate	K
3.925926e-01	1
3.518519e-01	2
3.407407e-01	3
3.333333e-01	4
3.555556e-01	5
3.296296e-01	6
3.481481e-01	7
3.518519e-01	8
3.444444e-01	9
3.333333e-01	10

The corresponding MATLAB script has been attached to the submitted file in directory *src*.

5 Computer Project 2

A one-dimensional Gaussian distribution with $\mu_1 = 0$ and $\sigma_1^2 = 1$ and a two dimensional Gaussian distribution with $\mu_2 = \begin{bmatrix}0\\0\end{bmatrix}$ and $\Sigma = I$ have been assumed. 30 samples from both distribution have been drawn and used for all following experiments. Note that the generated datasets have been the same in all experiments.

Parzen Kernel :

In the following experiments we are going to analyse the effects of kernel function in Parzen window density estimation. We have examined 3 different kernel functions in Parzen window density estimation. equations (8) to (10) represent these kernels for one dimensional density estimation.

$$\phi_1(X) = \begin{cases} 0.5 & |X| < |\frac{h}{2}| \\ 0 & otherwise \end{cases} \tag{8}$$

$$\phi_2(X) = \begin{cases} 0.5 & G(X|\mu_1, \frac{\sigma_1^2}{10}) >= G(1|\mu_1, \frac{\sigma_1^2}{10}) \\ 0 & otherwise \end{cases} \tag{9}$$

$$\phi_3(X) = \begin{cases} 0.5 & G(X|\mu_1, \sigma_1^2) >= G(1|\mu_1, \sigma_1^2) \\ 0 & otherwise \end{cases} \tag{10}$$

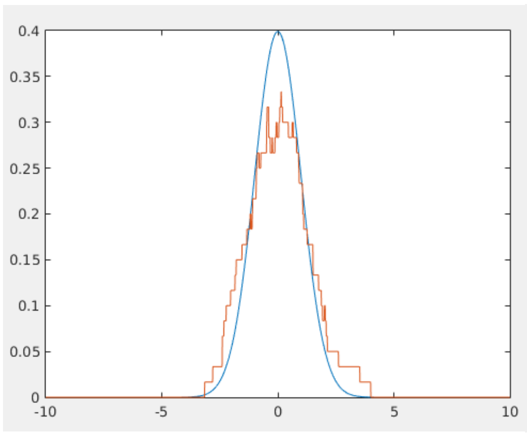
Note that the only difference between $\phi_2(X)$ and $\phi_3(X)$ is their variances. Figure 7 displays the estimated density functions along with real density curve.

As it is clearly obvious in figure 7, both form of kernels (Gaussian kernels $\phi 2$ and phi_3 and linear kernel ϕ_1) are able to estimate density function. The decrements in variances of Gaussian kernels, make estimations more spiky.
Assumed two dimensional Gaussian distribution has been shown in figure 8 which is used in all two dimensional experiments as the density to be estimated. Figure 9 represents the results of 2D density estimation using Parzen window with different kernels (11) and (12).

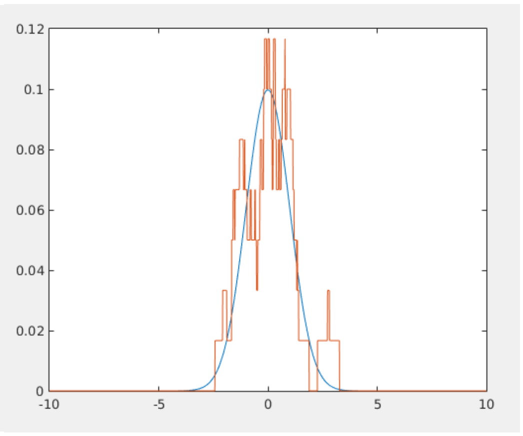
$$\phi_4(X) = \begin{cases} 0.5 & ||X||_2^2 < |\frac{h}{2}| \\ 0 & otherwise \end{cases} \tag{11}$$

$$\phi_5(X) = \begin{cases} 0.5 & G(X|\mu_2, \Sigma) >= G(\begin{bmatrix}1\\1\end{bmatrix}|\mu_2, \Sigma) \\ 0 & otherwise \end{cases} \tag{12}$$

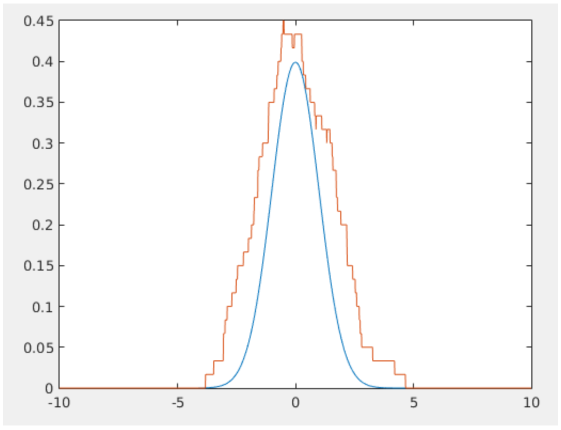
Similar to one-dimensional case, both types of kernels are able to estimate density function. In addition we can see in small number of given dataset, the



(a) Estimated density using $\phi_1(X)$



(b) Estimated density using $\phi_2(X)$



(c) Estimated density using $\phi_3(X)$

Figure 7: Effect of Parzen window kernel function to density estimation

normal kernel can make a better estimation of a originally normal distribution.

Parzen Bandwidth :

In the following experiments we are going to analyse the effects of bandwidth in Parzen window density estimation. The used kernel function in estimation of one-dimensional density in all cases of bandwidth, is $\phi_3(X)$. Figure 10 displays the estimation results with different bandwidths in Parzen window method.

As it is clear in the figure 10, the smoothness of estimated density function increases with increasing bandwidth but the accuracy decreases. Also in lower bandwidths, overfitting to the given dataset is obvious.
Also the results of density estimation in 2D case is similar to 1D density. Fig-

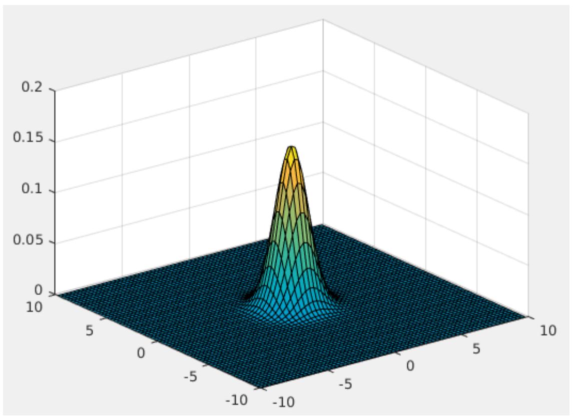
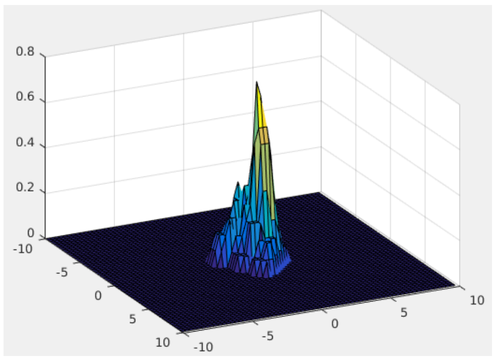
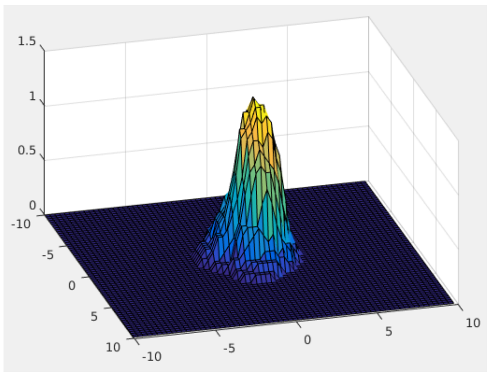


Figure 8: Assumed 2D Gaussian distribution to be estimated.



(a) Estimated density using ϕ_4



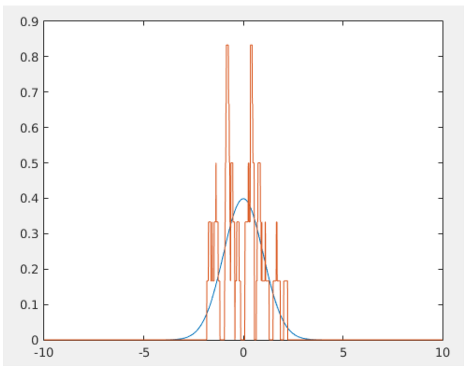
(b) Estimated density using ϕ_5

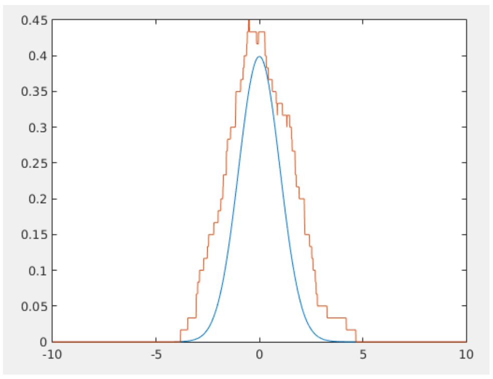
Figure 9: Estimated 2D density function using Parzen window using different kernels.

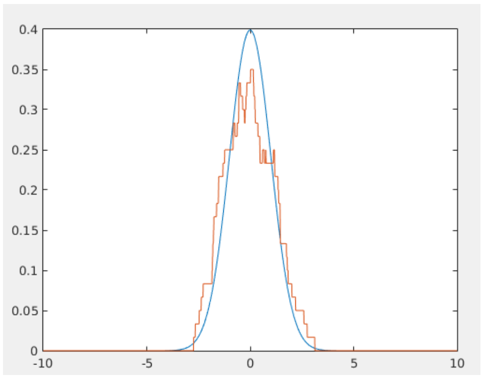
ure 11 displays the results from estimating 2D density displayed in figure 8. In this experiment the kernel function ϕ_4 has been used for the sake of simplicity. Similar to one-dimensional case, with increases in bandwidth, estimated density function gets coarser. All results and conclusions from one-dimensional case is compatible with two dimensional case, too.

K in K-NN :

In the following experiments we are going to analyse the effects of different Ks in K-NN model. Increasing K in the K-NN model, tends to search for neighbours in a larger region and makes model coarser. This fact is obvious in figure 12 which illustrates the estimation results for one-dimensional density with different values for K.


(a) Estimated density with $H = 0.1$


(b) Estimated density with $H = 0.5$


(c) Estimated density with $H = 1$

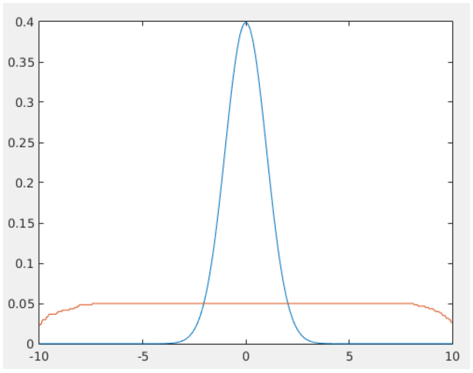
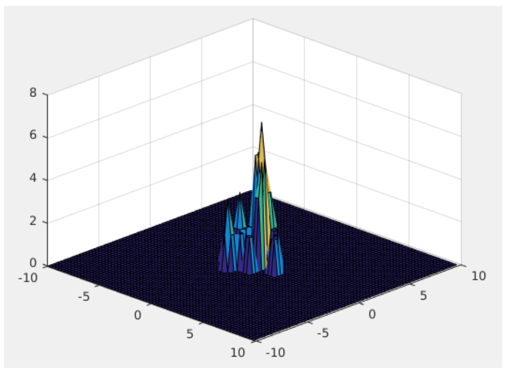

(d) Estimated density with $H = 10$

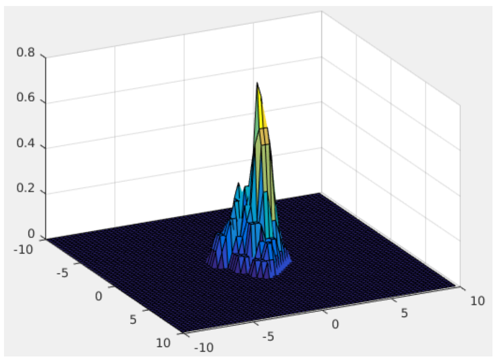
Figure 10: Estimated densities using Parzen window with different bandwidths.

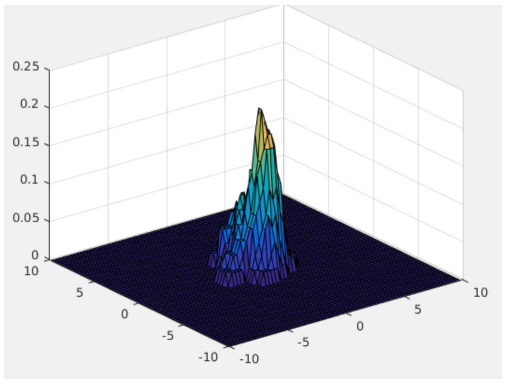
As it is expected, with lower values for K, the model becomes more spiky and we get near to zero estimates in a lot of wrong regions. Increasing K, results in coarser model. In the coarser models a very smooth estimation gained. The drawback of larger Ks is time wasting. More Ks result in needing more time for searching neighbours and wasted more memory.
The results of 2D density estimation are similar to 1D case. Figure 13 displays the density estimation for a 2D distribution.

Similar to 1D case, the larger K, the coarser model. Effects of outliers and noisy samples will be covered better when K is larger so the resulted model is smoother and has more generalizability. When K is extremely small, zero estimates appear and points near to a noisy sample will be misclassified and the model will be overfitted to training samples set.

Histogram bins :
In the following experiments we are going to analyse the effects of number of


(a) Estimated 2D density with $H = 0.1$


(b) Estimated 2D density with $H = 0.5$


(c) Estimated 2D density with $H = 1$

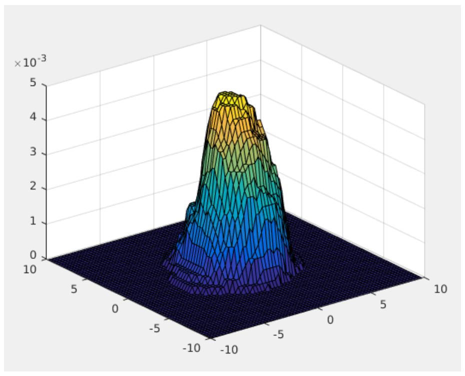
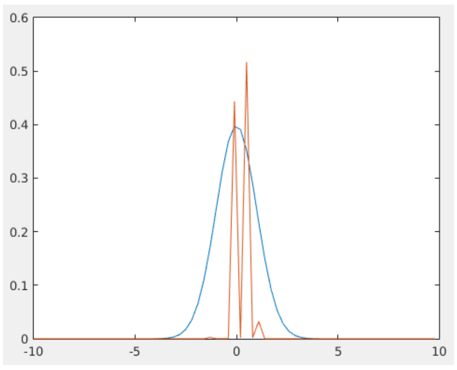

(d) Estimated 2D density with $H = 10$

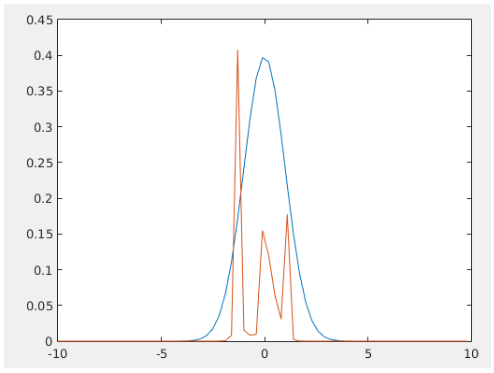
Figure 11: Estimation results for 2D density estimation using Parzen window with different bandwidths.

bins in histogram model. When number of bins is small, the averaging regions are larger and the model is coarser. While increasing number of bins, the area of each bin will continuously be decreased and model becomes finer. In general finer models are more local and make local decisions. Also they become more overfitted to training dataset and therefore has more accuracy.
Figure 14 illustrates histogram density estimation with different bin counts for estimating one dimensional distribution.

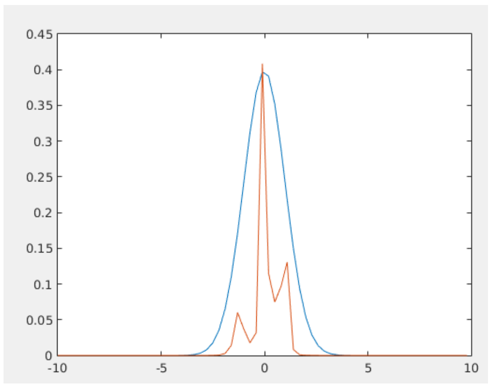
As it is clearly obvious from figure 14, the estimated density gets finer while increasing number of bins. Model with 20 bins has more spiky points ans has zero estimates in wrong regions and is more overfitted to training dataset.
The effect of bin counts in 2D density estimation is exactly the same as 1D case. As it is illustrated in figure 15 the behaviour or histogram model while increasing bin counts is just like previous experiment and model with more bin counts are finer models.



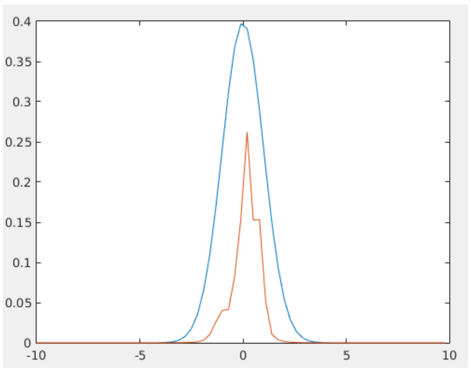
(a) Estimated density with $K = 1$



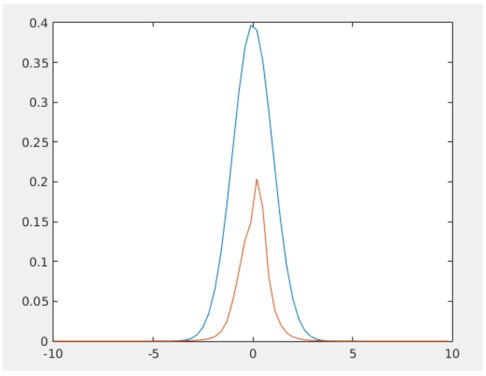
(b) Estimated density with $K = 3$



(c) Estimated density with $K = 5$

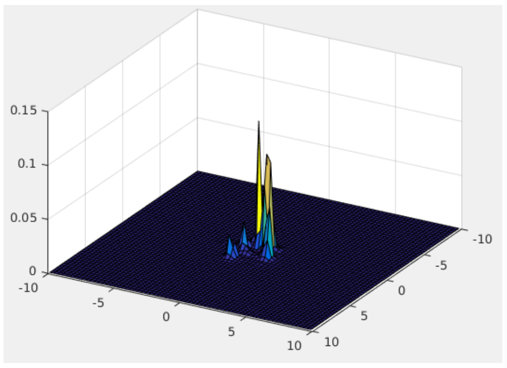


(d) Estimated density with $K = 10$

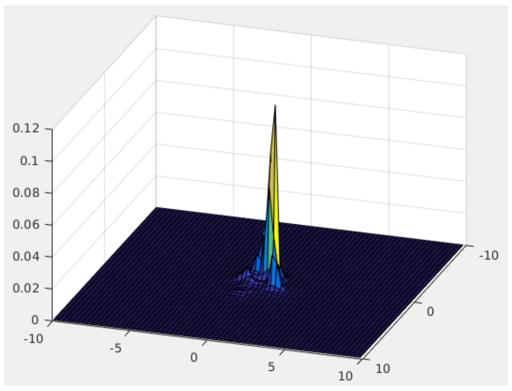


(e) Estimated density with $K = 20$

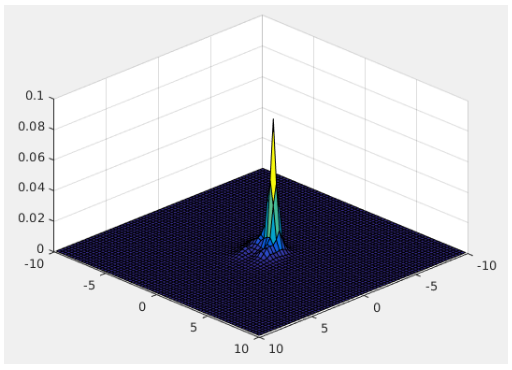
Figure 12: Estimation results for 1D density estimation using K-NN with different Ks.



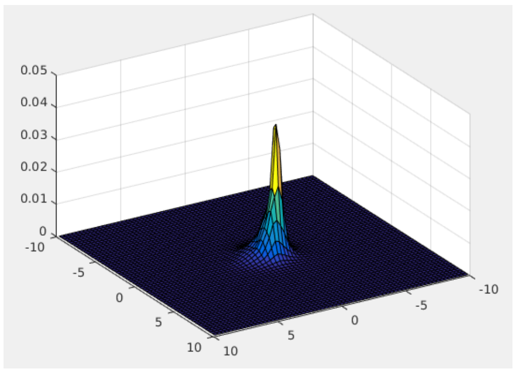
(a) Estimated density with $K = 1$



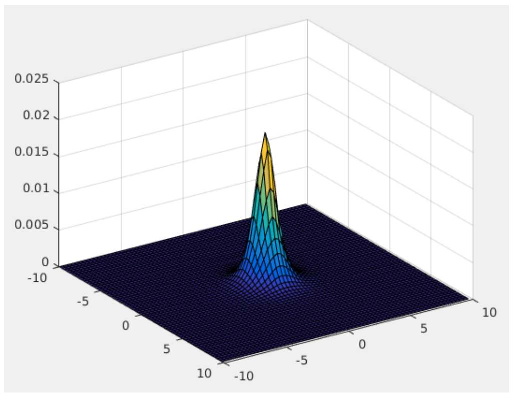
(b) Estimated density with $K = 3$



(c) Estimated density with $K = 5$

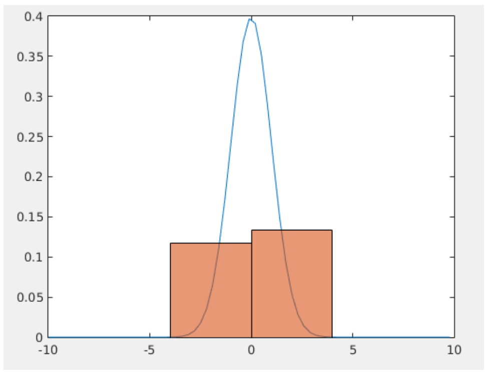


(d) Estimated density with $K = 10$

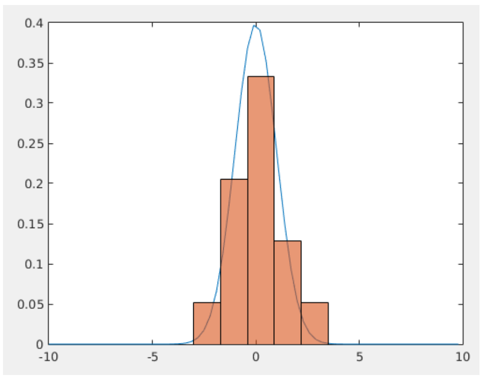


(e) Estimated density with $K = 20$

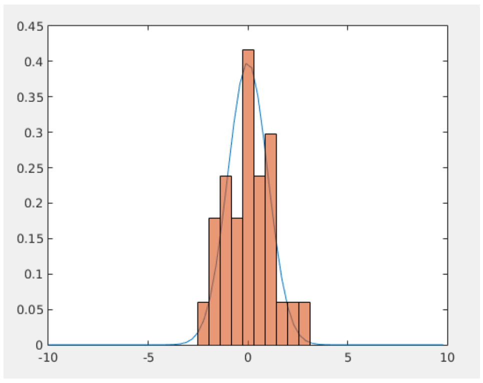
Figure 13: Estimation results for 2D density estimation using K-NN with different Ks.



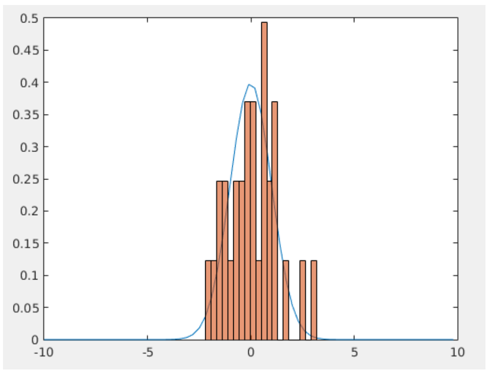
(a) Estimation using 2 bins



(b) Estimation using 5 bins

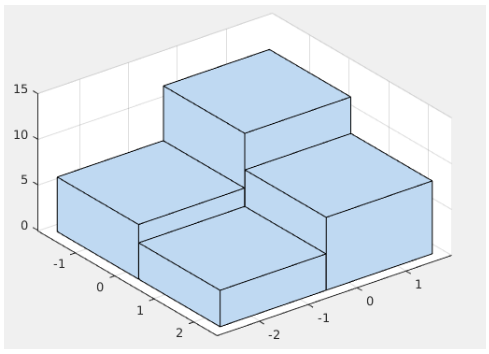


(c) Estimation using 10 bins

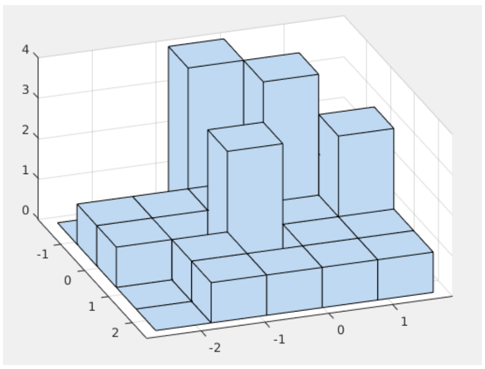


(d) Estimation using 20 bins

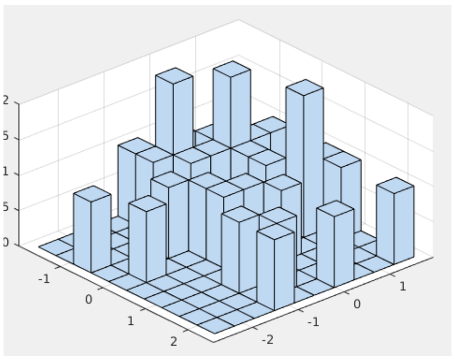
Figure 14: Estimated 1D density function using histogram density estimation with different bin counts.



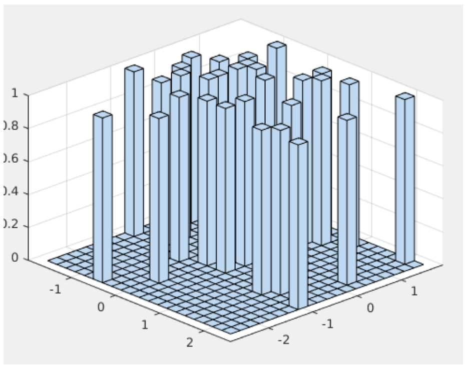
(a) Estimation using 2 * 2 bins



(b) Estimation using 5 * 5 bins



(c) Estimation using 10 * 10 bins



(d) Estimation using 20 * 20 bins

Figure 15: Estimated 2D density function using histogram density estimation with different bin counts.

References

Duda, R. O., Hart, P. E., and Stork, D. G. (2012). *Pattern classification*, page 24. John Wiley & Sons.