Harrison Tietze Multivariate Analysis Project

Professor Marchev Fall 2016

1 Kernel Density estimation (KDE)

1.1 Introduction

KDE is a non-parametric technique used to estimate the probability density function of a random variable. Suppose $(x_1, \ldots, x_n)$ is an independent and identically distributed sample drawn from some distribution with an unknown density f. We are interested in estimating the shape of this function f. Let's consider how this is derived in the univariate case:

Let x_0 be a new observation from our distribution. Choose a small "bandwidth" value h to construct an interval centered around x_0: (x_0-h, x_0+h). Now we can estimate $f(x_0)$ by the proportion of the sample contained in the interval, denoted $N(x_0)/n$, which represents the total number of the sample observations found in the interval divided by sample size. Here is how we estimate the density:

$$\frac{N(x_0)}{n} = \hat{P}(|x - x_0| < h) \approx 2h\hat{f}(x_0)$$

$$\Rightarrow \hat{f}(x_0) = \frac{N(x_0)}{2hn}$$

At this point, we need to express $N(x_0)$ as a function of the the entire sample. This is done by expressing $N(x_0)$ as the sum of kernel functions. The kernel function essentially takes an observation from the sample and returns its contribution towards $N(x_0)$. This can be achieved with a simple characteristic function that returns 1 if the observation is in the interval, and 0 if not. However, it is unattractive because it produces a step-function density that does not generalize well to the multivariate case. For this reason, we will focus on using a "smooth" kernel. Define the gaussian kernel using the standard normal distribution:

$$K(u) = \frac{1}{\sqrt{2\pi}} e^{-\frac{u^2}{2}}$$

And estimate the density with:

$$\hat{f}(x_0) = \frac{1}{hn} \sum_{i=1}^{n} K(\frac{x_0 - x_i}{h})$$

This way, points further outside the interval contribute less to the density, while points closest to x_0 contribute the most. The sum of the kernels approximates half the number of sample observations found in the interval. Intuitively, this is because the maximum value of the kernel is around $.4$

instead of 1 for the characteristic function. To compensate, we multiple the sum by 2, but this cancels out with the 2 we had in the denominator to represent the interval length of $2h$.
Note that using a gaussian kernel does not imply normality about the underlying distribution. KDE can be used to estimate any unknown distribution.

Now we will extend this to the multivariate case, this this is the formula we will use to analyze the data. Suppose we have a sample $x_1 \dots x_n$ where $x \in \mathbb{R}^p$ and we want to estimate the density of $x_0 = x(x_{01}, \dots, x_{0p})$. We use a multivariate normal kernel:

$$\boxed{\hat{f}(x_0) = \frac{1}{nh^p |S_{pl}|^{\frac{1}{2}}} \sum_{i=1}^{n} \exp\{-(x_0 - x_i)' S_{pl}^{-1} (x_0 - x_i)/(2h^2)\}}$$

Also a note about the bandwidth parameter: The size of h determines how much each observation contributes to the density estimate. If h is too small, $\hat{f}(x_0)$ has a peak at each x_i , and if h is too large, $\hat{f}(x_0)$ is almost uniform. Therefore, the value chosen for h must depend on the sample size n to avoid too much or too little smoothing; the larger the sample size, the smaller h should be. In practice, we could try several values of h and check the resulting error rates.

1.2 KDE for classification analysis

To use the kernel method of density estimation in classification, we can apply it to each group to obtain$\hat{f}(x_0|G_i)$ for $i = 1, \ldots, k$, where x_0 is a p-dimensional vector of unknown group membership. Assuming a uniform prior for the groups, the classification rule then becomes: Assign x_0 to the group G_i for which $\hat{f}(x_0|G_i)$ is maximum.

KDE is an advantageous classifier when suspect the covariance matrices of the groups are different. If the covariances are equal, KDE will probably be outperformed by a more biased method such as linear classification.

KDE classification

Description of Data and Task

For the classification task I will be using the Root data from the Rencher textbook. The data describes 4 measurements taken on tree roots: its girth after 4 years, its growth after 4 years, its girth after 15 years, and its weight after 15 years. There are 6 groups called "stocks" and each one contains 8 observations. The goal is to build a KDE that can classify an root of unknown stock into the proper group with a decent error.

The KDE algorithm

I begin by importing the data and coding the KDE algorithm using a functional approach:

```
#import data
data=read.table("T6_2_ROOT.DAT",header=F)
dictionary=c("Stock","Girth4","Growth4","Girth15","Wt15")
colnames(data)=dictionary
root.group=data$Stock
root.data=data[,2:ncol(data)]

#initialize variables
n=nrow(root.data)
p=ncol(root.data)
S=cov(root.data)
invS=solve(S)
h.vals=c(.2,.5,.9,1,1.15,1.5,2,3,5)

#data frame by group
root.group.data=function(k)
{return(root.data[root.group==k,])}



#gaussian coefficient
const=function(h){return(1/(n*h^p*det(S)^(1/2)))}

#gaussian kernel
kernel=function(y,h,k)
{
  weight=0
  for( i in 1:8){weight= weight+
  exp(-as.numeric(as.matrix(y-as.matrix(root.group.data(k)[i,]))
                  %*%invS%*%t((as.matrix(y-as.matrix(root.group.data(k)[i,])))))/(2*h^2))
  }
  return(weight)
}

#KDE estimator
KDE=function(y,h,k)
{
  fy.hat=const(h)*kernel(y,h,k)
return(fy.hat)
  }

#KDE classifier
KDE.classifier=function(y,h)
{
probs=c()
for(k in 1:6)
{
  probs[k]=KDE(y,h,k)
}

return(which.max(probs))
}




#adjusting bandwidth
KDE.bandwidth=function(h){
classification.results=c()
for(i in 1:n){
  y=root.data[i,]
   classification.results[i]=KDE.classifier(y,h)
}
class.table=matrix(classification.results,nrow=8,byrow = F)
return(class.table)
}

#misclassification error
error=function(class.table){
  wrong.count=0
  for(i in 1:ncol(class.table)){
    for(j in 1:nrow(class.table)){
      wrong.count=wrong.count+ifelse(class.table[j,i]==i,0,1)
    }
  }
  error.rate=wrong.count/(ncol(class.table)*nrow(class.table))
  return(error.rate)
}
```

Classification tables

We can observe how well the KDE classifier does for different values of h: The classification matrix shows the 6 groups as columns, and each entry is the predicted group. The misclassification error is calculated by the proportion of misses.

```
KDE.bandwidth(.8)
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6]
## [1,]    1    2    1    4    5    6
## [2,]    1    2    3    4    5    6
## [3,]    1    2    5    1    5    6
## [4,]    1    2    3    4    5    6
## [5,]    1    2    3    4    5    6
## [6,]    1    2    3    4    5    6
## [7,]    1    2    3    4    5    6
## [8,]    1    2    3    4    5    6
```

```
error(KDE.bandwidth(.8))
```

```
## [1] 0.0625
```

```
KDE.bandwidth(1)
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6]
## [1,]    1    2    1    4    5    1
## [2,]    1    1    3    3    2    6
## [3,]    1    2    5    1    5    6
## [4,]    1    2    3    4    5    6
## [5,]    1    2    3    4    6    6
## [6,]    1    5    3    4    5    1
## [7,]    4    3    3    4    5    1
## [8,]    1    2    3    4    5    6
```

```
error(KDE.bandwidth(1))
```

```
## [1] 0.2708333
```

```
KDE.bandwidth(1.2)
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6]
## [1,]    1    5    1    1    5    1
## [2,]    1    1    3    3    2    6
## [3,]    1    2    5    1    5    6
## [4,]    1    2    3    4    5    6
## [5,]    1    2    3    4    6    6
## [6,]    1    5    3    4    5    1
## [7,]    4    3    3    4    5    1
## [8,]    1    2    3    4    5    6
```

```
error(KDE.bandwidth(1.2))
```

```
## [1] 0.3125
```

Discussion

As the bandwidth parameter decreases, the training error will go to 0 and the estimator will be overfitting the data. Around Below .8, the training error stabilizes at 0, indicating the model is too flexible. If h is too large, the kernel will be over-smooth and uninformative. In this case we can choose h=1, as it has a decent apparent error rate and still classifies well. We can conclude that using h=1, our KDE could classify an an unknown root into 1 of the 6 groups just by looking at those 4 features, with an apparent error rate of about 27%.

Sources:

Methods of Multivariate Analysis, 2nd Edition, Alvin C. Rencher Wikipedia: Kernel Density Estimators