LAB 3 BLOCK 1: KERNEL METHODS AND NEURAL NETWORKS

Zuxiang Li

12/16/2019

1. KERNEL METHODS

Implement a kernel method to predict the hourly temperatures for a date and place in Swe- den. To do so, you are provided with the files stations.csv and temps50k.csv. These files contain information about weather stations and temperature measurements in the stations at different days and times. The data have been kindly provided by the Swedish Meteorological and Hydrological Institute (SMHI).

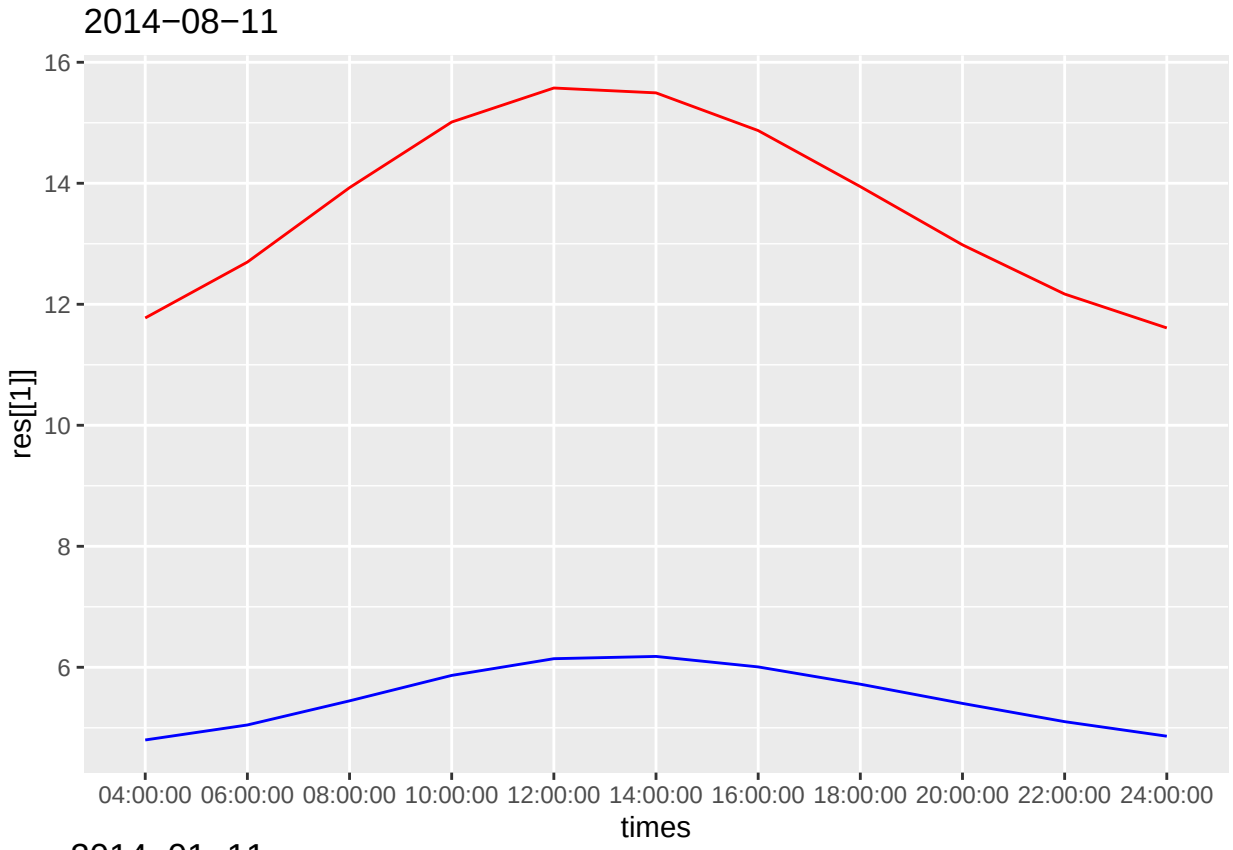
You are asked to provide a temperature forecast for a date and place in Sweden. The forecast should consist of the predicted temperatures from 4 am to 24 pm in an interval of 2 hours. Use a kernel that is the sum of three Gaussian kernels:

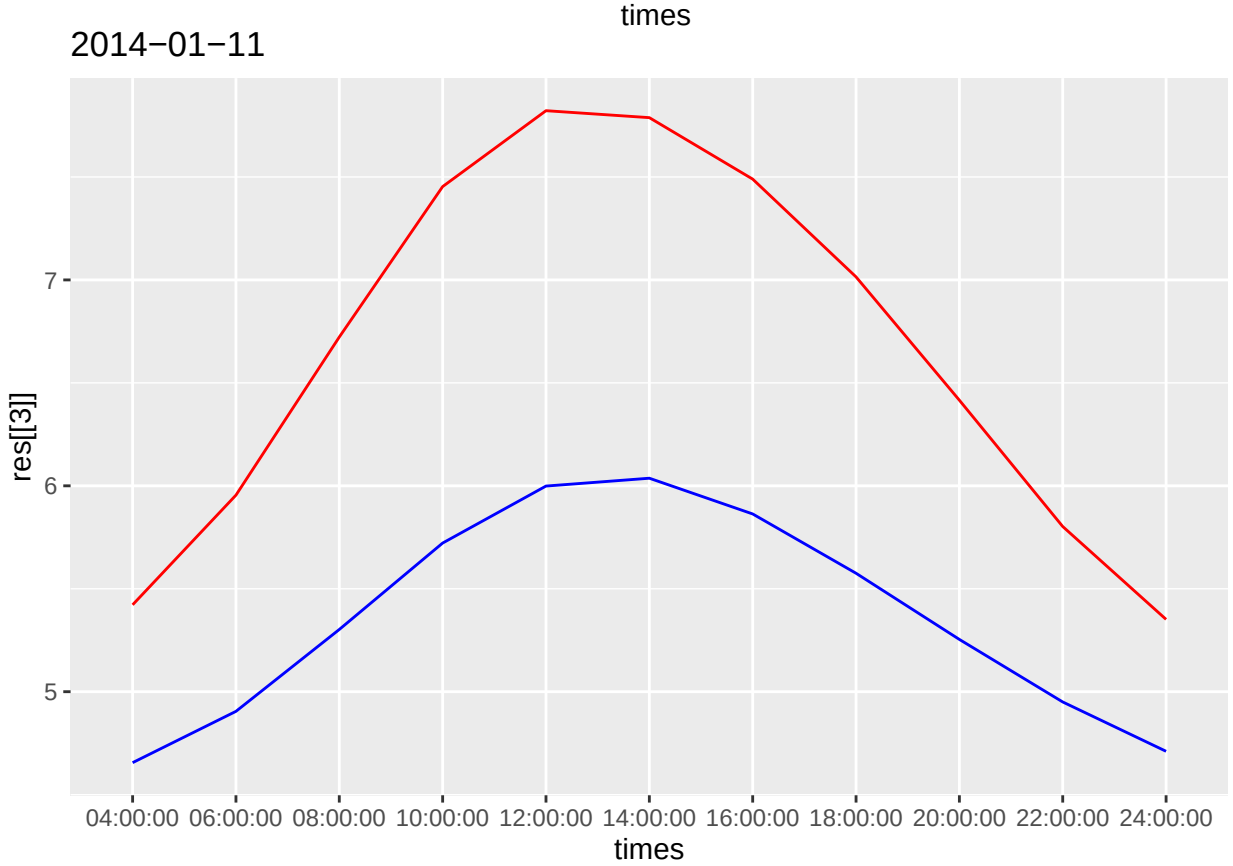
- The first to account for the distance from a station to the point of interest.
- The second to account for the distance between the day a temperature measurement was made and the day of interest.
- The third to account for the distance between the hour of the day a temperature mea- surement was made and the hour of interest.

Choose an appropriate smoothing coefficient or width for each of the three kernels above. Answer to the following questions:

- Show that your choice for the kernels' width is sensible, i.e. that it gives more weight to closer points. Discuss why your of definition of closeness is reasonable.
- Instead of combining the three kernels into one by summing them up, multiply them. Compare the results obtained in both cases and elaborate on why they may differ.

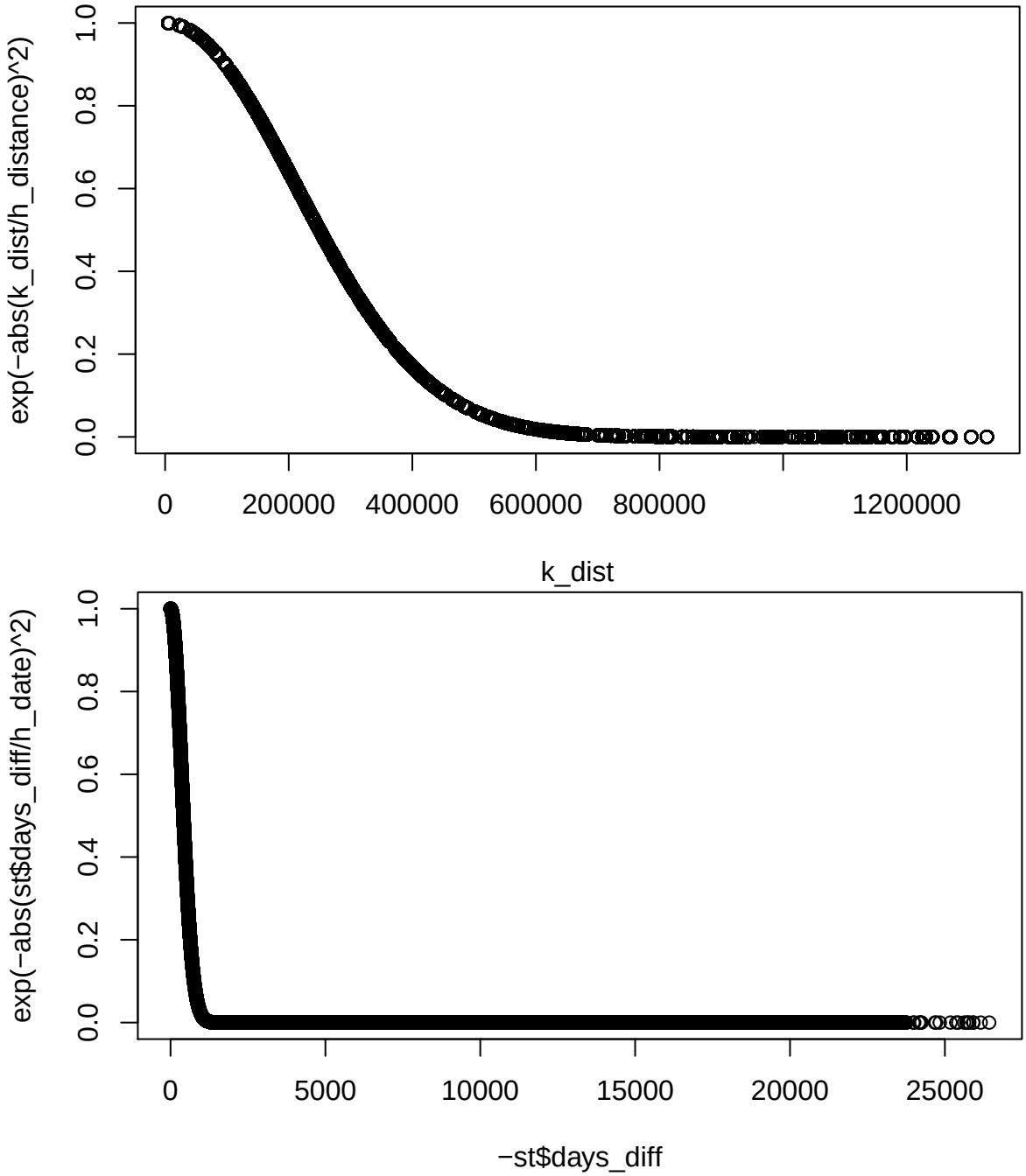
Note that the file temps50k.csv may contain temperature measurements that are posterior to the day and hour of your forecast. You must filter such measurements out, i.e. they cannot be used to compute the forecast. Feel free to use the template below to solve the assignment.

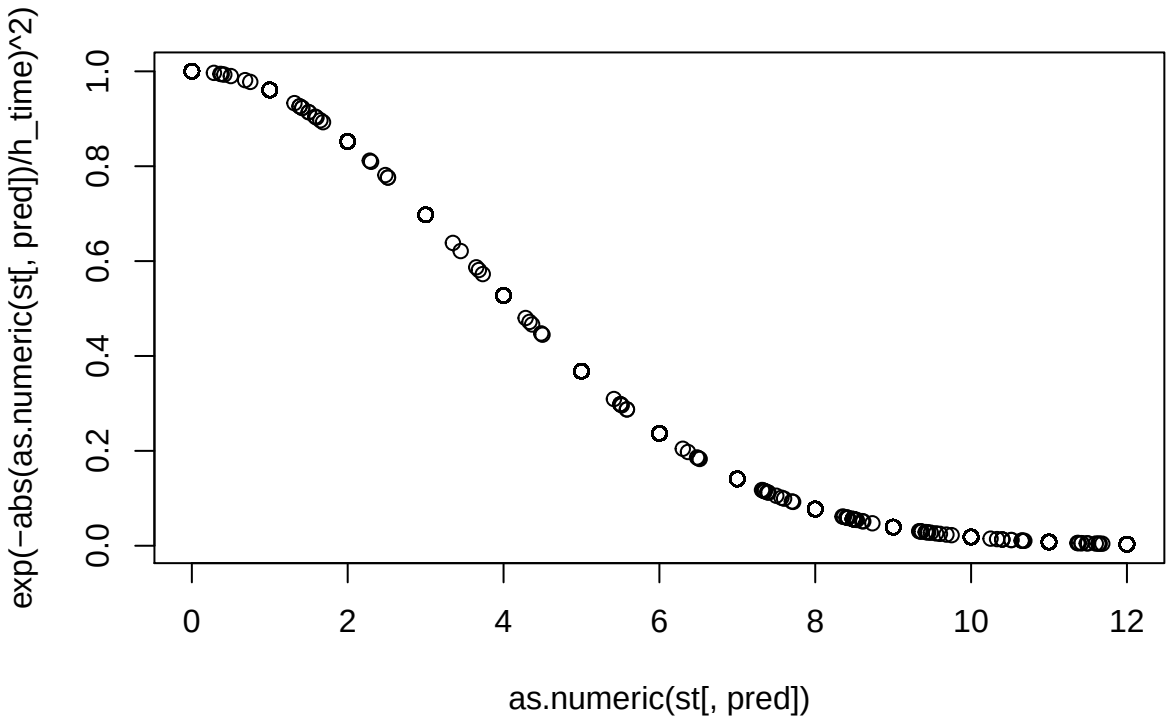




We select 58.397,15.578 as location, 2014-08-11 and 2014-01-11 as date. Red line is the curve of multiplication

kernel, blue line is summation kernel. In both plots, the trend of all lines seem normal, but for multiplication kernel it get better result comparing to the summation one since there should be a significant change in tempreature during summer and winter. As we can see the former one got above 12 degrees in summer and 5 or 7 degrees in winter, but for latter one, not much changes in the tempreature. The reason for this might because the summation kernel strengthen the dependency between location, date and times, for multiplication kernel it's not.





Here we select the kernel width by drawing the gaussian kernel, it is clear to see the closer points have a stronger effect on the result compare to the points in the tail. With my intuition, the h_distance, h_date, h_time should be 600000 meters, 500 days and 5 hours.

2. SUPPORT VECTOR MACHINES

Use the function ksvm from the R package kernlab to learn a SVM for classifying the spam dataset that is included with the package. Consider the radial basis function kernel (also known as Gaussian) with a width of 0.05. For the C parameter, consider values 0.5, 1 and 5. This implies that you have to consider three models.

- Perform model selection, i.e. select the most promising of the three models (use any method of your choice except cross-validation or nested cross-validation).
- Estimate the generalization error of the SVM selected above (use any method of your choice except cross-validation or nested cross-validation).
- Produce the SVM that will be returned to the user, i.e. show the code.
- What is the purpose of the parameter C ?

```
##
## Support Vector Machine object of class "ksvm"
##
## SV type: C-svc  (classification)
##  parameter : cost C = 0.5
##
## Gaussian Radial Basis kernel function.
##  Hyperparameter : sigma =  0.05
##
```

```
## Number of Support Vectors : 1497
##
## Objective Function Value : -439.4751
## Training error : 0.04837
##          pred
## test_y    nonspam spam
##   nonspam     523   18
##   spam         66  314
##
## True Positive Rate: 0.8879457
## True Negative Rate: 0.9457831
## False Positive Rate: 0.05421687
## False Negative Rate: 0.1120543
## Misclassfication Rate: 0.09120521
##
## Support Vector Machine object of class "ksvm"
##
## SV type: C-svc  (classification)
##  parameter : cost C = 1
##
## Gaussian Radial Basis kernel function.
##  Hyperparameter : sigma =  0.05
##
## Number of Support Vectors : 1396
##
## Objective Function Value : -657.783
## Training error : 0.041576
##          pred
## test_y    nonspam spam
##   nonspam     521   20
##   spam         56  324
##
## True Positive Rate: 0.9029463
## True Negative Rate: 0.9418605
## False Positive Rate: 0.05813953
## False Negative Rate: 0.09705373
## Misclassfication Rate: 0.082519
##
## Support Vector Machine object of class "ksvm"
##
## SV type: C-svc  (classification)
##  parameter : cost C = 5
##
## Gaussian Radial Basis kernel function.
##  Hyperparameter : sigma =  0.05
##
## Number of Support Vectors : 1306
##
## Objective Function Value : -1661.591
## Training error : 0.020109
##          pred
## test_y    nonspam spam
##   nonspam     519   22
##   spam         52  328
```

```
##
## True Positive Rate: 0.9089317
## True Negative Rate: 0.9371429
## False Positive Rate: 0.06285714
## False Negative Rate: 0.0910683
## Misclassfication Rate: 0.08034745
```

First we devide the data into train(80%) and test(20%). Then we train the model using train data with width=0.5 and different C values(0.5,1,5). From the result we can see when C increases, the misclassfication rate for test is decreasing.

Parameter C here can be viewed as a regularization parameter, with a larger C it reduces the misclassfication error, with a smaller C the model tended to be a low-complexity solutions.

In this case, the model should be

```
mod3<-ksvm(type~.,train,type="C-svc",kernel="rbfdot",kpar=list(sigma=0.05),C=5)
```

Appendix

1.

```
set.seed(1234567890)
library(geosphere)
library(readr)
stations <- read_csv("data/stations.csv")
temps <- read_csv("data/temps50k.csv")

  times <- c("04:00:00","06:00:00","08:00:00","10:00:00","12:00:00","14:00:00","16:00:00","18:00:00","20

weather_predict<-function(date,lat,lon,p=FALSE){

  h_distance <-300000 # These three values are up to the students
  h_date <-120
  h_time <-5



  temp <- vector(length=length(times))

  st <- merge(stations,temps,by="station_number")

  st<-st[,c(-3,-6,-7,-8,-12)]
  st$date<-as.Date(st$date)
  date<-as.Date(date)
  st$days_diff<-as.numeric(difftime(st$date,date,units = "days"))
  idx<-which(st$days_diff>=0)
  if(length(idx)!=0){
    st<-st[-idx,]
  }

  time1<-strptime(paste(st$time),"%H:%M:%S")
  time2<-strptime(paste(times[1]),"%H:%M:%S")
  time_diff<-as.numeric(abs(difftime(time1,time2,units = "hours")))
  time_diff<-ifelse(time_diff<12,time_diff,24-time_diff)

  df<-as.data.frame(time_diff)

  for(i in 2:length(times)){
    time1<-strptime(paste(st$time),"%H:%M:%S")
    time2<-strptime(paste(times[i]),"%H:%M:%S")
    time_diff<-as.numeric(abs(difftime(time1,time2,units = "hours")))
    time_diff<-ifelse(time_diff<12,time_diff,24-time_diff)
    tdf<-as.data.frame(time_diff)
    df<-cbind(df,tdf)
  }

  names(df)<-times
  st<-cbind(st,df)

  # Students' code here
```

```r
distance_kernel<-function(lat,lon,p=FALSE){
  k_dist<-distHaversine(st[,3:4],c(lat,lon))
  if(p==TRUE){
  plot(k_dist,exp(-abs(k_dist/h_distance)^2))
  }
  exp(-abs(k_dist/h_distance)^2)
}

date_kernel<-function(p=FALSE){
  res=exp(-abs(st$days_diff/h_date)^2)
  if(p==TRUE){
   plot(-st$days_diff,exp(-abs(st$days_diff/h_date)^2))
  }
  return(res)
}

time_kernel<-function(pred,p=FALSE){
  res=exp(-abs(as.numeric(st[,pred])/h_time)^2)
  if(p==TRUE){
    plot(as.numeric(st[,pred]),exp(-abs(as.numeric(st[,pred])/h_time)^2))
  }
  return(res)
}

sum_kernel<-function(pred,p=FALSE){
  sum_k<-distance_kernel(lat,lon,p)+date_kernel(p)+time_kernel(pred,p)
  res<-sum(sum_k*st$air_temperature)/sum(sum_k)
  return(res)
}

mul_kernel<-function(pred,p=FALSE){
  sum_k<-distance_kernel(lat,lon,p)*date_kernel(p)*time_kernel(pred,p)
  res<-sum(sum_k*st$air_temperature)/sum(sum_k)
  return(res)
}

# kernel test



temp<-c()
for(i in 1:length(times)){
  temp<-c(temp,sum_kernel(times[i]))
}
temp

temp_mul<-c()
for(i in 1:length(times)){
  temp_mul<-c(temp_mul,mul_kernel(times[i]))
}
  # ggplot()+geom_line(aes(x=times,y=temp,group=1),col="blue")+
  # geom_line(aes(x=times,y=temp_mul,group=1),col="red")+
  # ggtitle(paste(date))
```

```
  #sum_kernel(times[1],p=TRUE)
  return(list(temp,temp_mul))
}

date1 <- "2014-08-11" #The date to predict

a <- 58.397 #Latitude of the point to predict
b <- 15.578 #Longitude of the point to predict

date2 <- "2014-01-11"
# weather_predict(date1,a,b)
# weather_predict(date2,a,b)
#res<-weather_predict(date1,a,b)
#res
res<-c()
for(i in c(date1,date2)){
  res<-c(res,weather_predict(i,a,b))
}

ggplot()+geom_line(aes(x=times,y=res[[1]],group=1),col="blue")+
  geom_line(aes(x=times,y=res[[2]],group=1),col="red")+
  ggtitle(paste(date1))

ggplot()+geom_line(aes(x=times,y=res[[3]],group=1),col="blue")+
  geom_line(aes(x=times,y=res[[4]],group=1),col="red")+
  ggtitle(paste(date2))
```

2

```
library(kernlab)
data("spam")

n=dim(spam)[1]
set.seed(1234567890)
id=sample(1:n, floor(n*0.8))
train=spam[id,]
test=spam[-id,]

mod1<-ksvm(type~.,train,type="C-svc",kernel="rbfdot",kpar=list(sigma=0.05),C=0.5)
mod2<-ksvm(type~.,train,type="C-svc",kernel="rbfdot",kpar=list(sigma=0.05),C=1)
mod3<-ksvm(type~.,train,type="C-svc",kernel="rbfdot",kpar=list(sigma=0.05),C=5)
test_x=test[,-58]
test_y=test[,58]

for(i in c(mod1,mod2,mod3)){
  pred=predict(i,test_x)
  cft<-table(test_y,pred)
  cat("\n")
  print(i)
  print(cft)
  cat("\nTrue Positive Rate:",cft[1,1]/(cft[1,1]+cft[2,1]))
```

```
  cat("\nTrue Negative Rate:",cft[2,2]/(cft[1,2]+cft[2,2]))
  cat("\nFalse Positive Rate:",cft[1,2]/(cft[1,2]+cft[2,2]))
  cat("\nFalse Negative Rate:",cft[2,1]/(cft[2,1]+cft[1,1]))
  mis_rate=1-((cft[1,1]+cft[2,2])/sum(cft))
  cat("\nMisclassfication Rate:",mis_rate)
  cat("\n")
}
```