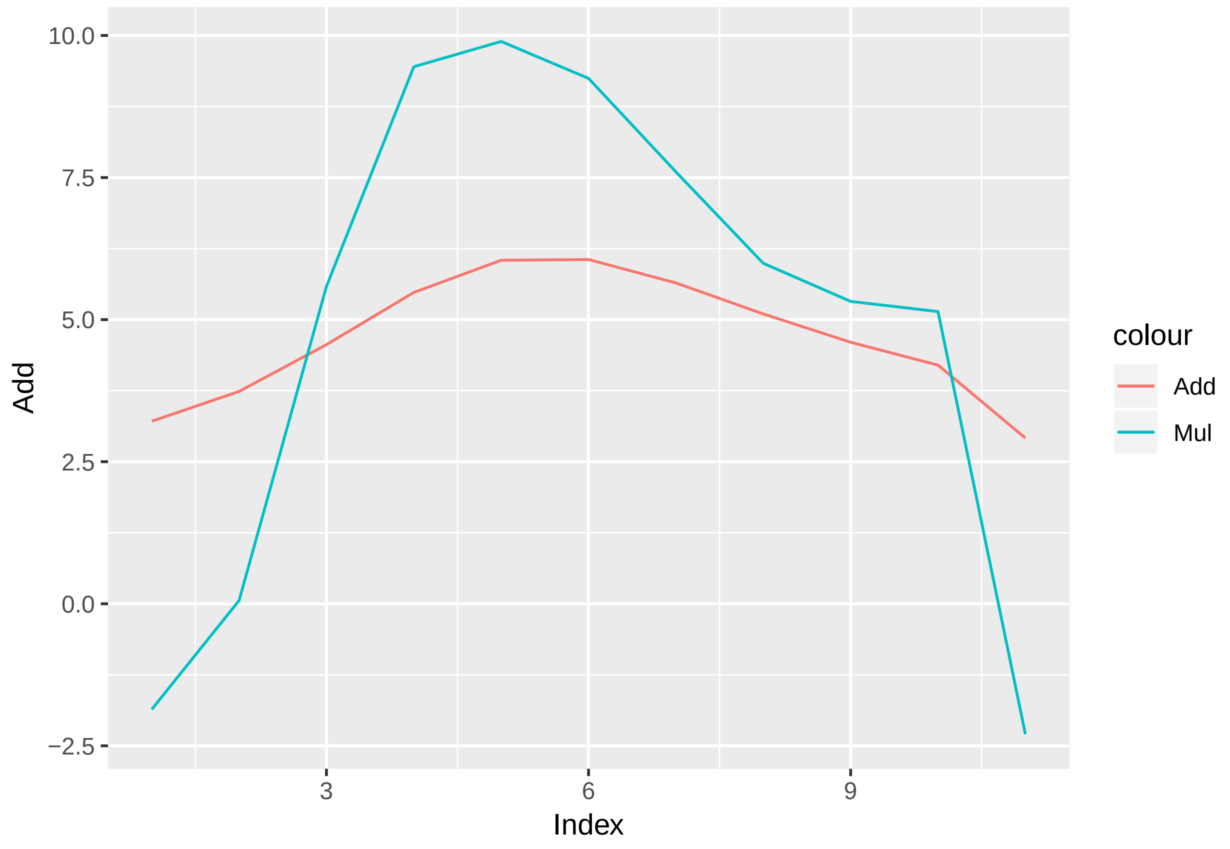
Lab3

Sridhar Adhikarla

18 December 2018

Contents

Assignment 1 - Kernel Methods (Temperatue Forecast)



These are the two predictions we are getting on Adding all the kernals and Multiplying all the kernals. They are pretty close to each other for this given parameter values of the kernal. This prediction is been made for the 4th of May 2011, from 4 in the morning to 24 at night for that day. The temperature predicted by both the kernals is almost similar, and the temperatures looked to be correct for a day in month of May, these are some of the reasons we selected these values for the parameters.

The parameter value for distance we chose 30000 meters, that is 30 kms. We chose this value as the temperature in an area would be similar for an approximate 30kms radius. This is the reason we gave "h_distance=30000". The function "distHaversine" gives the result in meters, so we had to specify the value in meters.

The parameter value for date we chose is 12 days. This means the temperature of the date we chose to predict for will be dependent on 12 days prior to that date. Those prior dates will be given higher weights for prediction of temperature. This seems reasonable for us as the tempreatre usually does not fluctuate much and is quiet similar along the 10 - 12 days margin.

The parameter value for time we chose is 4 hours. This means the temperature of the time of date we chose to predict for will be dependent on 4 hours prior to that time of the day. We thought the temperature value would not fluctuate much in 3-4 hours time period, it would be similar.

The multiplicative kernel peaks at the mid day and falls down again, but the additive kernel has a large standard deviation and is smooth in predictions. There are no sudden rise and fall in temperatures in a day according to the additive kernel, which seems right. Therefor we would say the additive kernel performs better than the multiplicative kernel.

But the additive kernel seems to give a lower bound on the temperatue and the multiplicative kernel gives a upper bound on the temperature prediction. So if we somehow take an average on the prediction of the kernal that could be a better predicion of temperature.

The prediction of the additive kernel moves smoothly because it is not much affected by sudden changes in temperature, as we are adding the values and averaging over it. But in case of multiplicative kernel the kernel gets affected a lot with the sudden changes in temperature and this is the reaon it has such sudden movements in the plot and is not smooth whereas the additive plot is smooth.

Assignment2 : Support Vector Machine

1. SVM with C=0.5,1,5

I have divided the data into train and valid in the ratio of 60%-40%. The ROC curve is based on train data sets and svm model for C=0.5,1 and 5 trained on train dataset.

```
## Model with C=0.5
```

```
## 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 : 1237
##
## Objective Function Value : -356.9024
## Training error : 0.055072
## Probability model included.
```

```
##
##
## Model with C=1
```

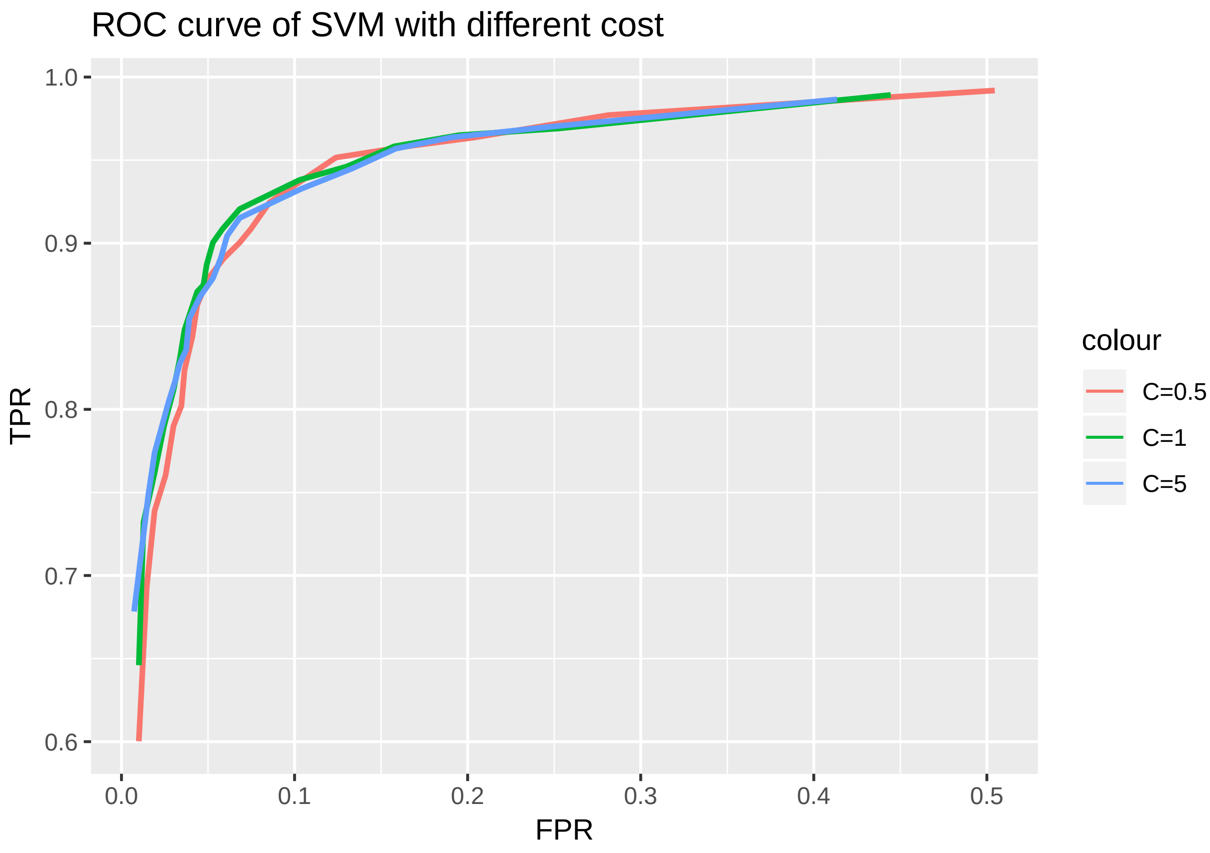
```
## Support Vector Machine object of class "ksvm"
##
## SV type: C-svc  (classification)
##  parameter : cost C = 1
##
## Gaussian Radial Basis kernel function.
##  Hyperparameter : sigma =  0.05
##
## Number of Support Vectors : 1134
##
## Objective Function Value : -535.0396
## Training error : 0.041304
## Probability model included.
```

```
##
##
## Model with C=0.5
```

```
## 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 : 1083
##
## Objective Function Value : -1288.147
## Training error : 0.02029
## Probability model included.
```

I have used ROC for model selection . Below is the ROC curve for model selection with C=0.5, 1 and 5.



2. Generalization Error

The generalization error is calculated on valid dataset. Even though, all the three model has high accuracy for classfying spam, I have chosen, C=0.5 to be a better model since it is robust. Even though this model has missclassifictaion error more than that of model with C=1 and C=5 C=0.5 but it is expected to perform better because of wider margin and number of missclassification might be less for future data. Also ROC curve shows that the missclassification error for C=0.5 is more but I think, it is robust.

```
##
##
## Confusion Matrix for C=0.5 :
##
##

##          Predicted
## Expected  nonspam spam
##   nonspam    1057   41
##   spam        126  617
```

```
##
##
## Missclassification Error with C=0.5: 0.09071157
```

3. Purpose of the parameter C

Parameter C signifies the cost for residuals. Higher cost would result in high variance and less bias. This means that higher the cost C the margins will become more narrow to decrease the missclassification rate.

Appendix

```r
knitr::opts_chunk$set(
    echo = FALSE,
    message = FALSE,
    warning = FALSE
)
library(kernlab)
library(tidyverse)
library(geosphere)
#Assignment 1
kernal_methods <- function(st, date, lat, lon, time_seq, h_date, h_time, h_distance){
  st$h_dist = abs(distHaversine(p1 = c(lon, lat), p2 = st[,c("longitude", "latitude")]))
  st$h_dist = exp(-(st$h_dist/h_distance)^2)
  st$h_date = as.numeric(difftime(date, st$date, units = c("days")))
  st$h_date = ifelse(st$h_date>0, st$h_date, 0)
  st = subset(st, st$h_date!=0)
  st$h_date = exp(-(st$h_date/h_date)^2)
  times = c()
  for(t in 1:length(time_seq)){
    d = as.Date(time_seq[t])
    c_time = format(time_seq[t], "%H:%M:%S")
    times = append(times, c(c_time))
    st[c_time] =  as.numeric(abs(difftime(strptime(paste(d, c_time),
                                                   "%Y-%m-%d%H:%M:%S"),
                                          strptime(paste(d, st$time),
                                                   "%Y-%m-%d%H:%M:%S"),
                                          units = c("hour"))))
    st[c_time] = exp(-(st[c_time]/h_time)^2)
  }
  temp = st[times]
  temp_add_d = temp + (st$h_date + st$h_dist)
  temp_mul_d = temp * (st$h_date * st$h_dist)
  temp_add_n = temp_add_d*st$air_temperature
  temp_mul_n = temp_mul_d*(st$air_temperature)

  temp_add = colSums(temp_add_n)/colSums(temp_add_d)
  temp_mul = colSums(temp_mul_n)/colSums(temp_mul_d)

  d = data.frame(Index = 1:length(times), Time = times, Add = temp_add, Mul = temp_mul)
  return(d)
}

set.seed(1234567890)
stations = read.csv("stations.csv")
temps = read.csv("temps50k.csv")
st <- merge(stations,temps,by="station_number")
#colnames(st)
h_distance <- 30000
h_date <- 12
h_time <- 4
lat <- 58.4274 # The point to predict (up to the students)
lon <- 14.826
date <- "2014-05-04" # The date to predict (up to the students)
start <- as.POSIXct(date)
interval <- 60
end <- start + as.difftime(1, units="days")
time_seq <- seq(from=start, by=interval*120, to=end)
time_seq = time_seq[3:length(time_seq)]

pred = kernal_methods(st, date, lat, lon, time_seq, h_date, h_time, h_distance)
ggplot(pred) + geom_line(aes(Index, Add, col="Add")) + geom_line(aes(Index, Mul, col="Mul"))
data("spam",package="kernlab")
spamdata <- spam

nr <- nrow(spamdata)
set.seed(12345)
id <- sample(1:nr,floor(0.6*nr))

spamdata_train <- spamdata[id,]
spamdata_test  <- spamdata[-id,]

#rbfdot is the radial basis function for the kernel type.
#C signifies the cost for residuals. Higher cost would result in
#high variance and less bias.

#With C=0.5
svm_model <- ksvm(type~., data=spamdata_train,kernel="rbfdot",kpar=list(sigma=0.05),
                  prob.model=TRUE,C=0.5)
predict1 <- predict(svm_model,newdata=spamdata_test,type="probabilities")

#With C=1
svm_model2 <- ksvm(type~., data=spamdata_train,kernel="rbfdot",kpar=list(sigma=0.05),
                   prob.model=TRUE,C=1)
predict2 <- predict(svm_model2,newdata=spamdata_test,type="probabilities")

#With C=5
svm_model3 <- ksvm(type~., data=spamdata_train,kernel="rbfdot",kpar=list(sigma=0.05),
                   prob.model=TRUE,C=5)
predict3 <- predict(svm_model3,newdata=spamdata_test,type="probabilities")

#Comaprison
#ROC curve, bootstrap, AIC, BIC,Jaccard similarity coefficient score
```

```r
cat("Model with C=0.5\n")
svm_model

cat("\n\nModel with C=1\n")
svm_model2


cat("\n\nModel with C=0.5\n")
svm_model3
#ROC Curve

origval<-ifelse(spamdata_test$type=="spam",1,0)
compare_data <- matrix(nrow=0,ncol=6)
colnames(compare_data) <- c("TPR_predc1","FPR_predc1",
                            "TPR_predc2","FPR_predc2",
                            "TPR_predc3","FPR_predc3")

for ( i in seq(0.05,0.95,0.05)) {

    prob_conv1  <- ifelse(predict1[,2]>i,1,0)
    prob_conv2  <- ifelse(predict2[,2]>i,1,0)
    prob_conv3  <- ifelse(predict3[,2]>i,1,0)

    TPR_predc1 <- sum(prob_conv1*origval) /sum(origval)
    FPR_predc1 <- (sum(prob_conv1)-sum(prob_conv1*origval))/sum(origval==0)

    TPR_predc2<- sum(prob_conv2*origval) /sum(origval)
    FPR_predc2 <- (sum(prob_conv2)-sum(prob_conv2*origval))/sum(origval==0)

    TPR_predc3 <- sum(prob_conv3*origval) /sum(origval)
    FPR_predc3 <- (sum(prob_conv3)-sum(prob_conv3*origval))/sum(origval==0)

    compare_data<-rbind(compare_data,c(TPR_predc1,FPR_predc1,
                        TPR_predc2,FPR_predc2,TPR_predc3,FPR_predc3))

}

compare_data <- as.data.frame(compare_data)
colnames(compare_data) <- c("TPR_predc1","FPR_predc1",
                            "TPR_predc2","FPR_predc2",
                            "TPR_predc3","FPR_predc3")
ggplot(compare_data) + geom_line(aes(x=FPR_predc1,y=TPR_predc1,color="C=0.5 ",size=I(1))) +
 geom_line(aes(x=FPR_predc2,y=TPR_predc2,color="C=1",size=I(1)))+
    geom_line(aes(x=FPR_predc3,y=TPR_predc3,color="C=5",size=I(1)))+
    geom_abline(intercept=0,slope=1)+
    xlab("FPR")+ylab("TPR")+ggtitle("ROC curve of SVM with different cost ")
svm_model <- ksvm(type~.,data=spamdata_train,kernel="rbfdot",kpar=list(sigma=0.05)
                  ,C=0.5)
predict <- predict(svm_model,newdata=spamdata_test)

conf_matrix <- table("Expected" = spamdata_test$type,"Predicted"=predict)

cat("\n\nConfusion Matrix for C=0.5 :\n\n ")
```

```
conf_matrix

missclas <-  sum(conf_matrix[1,2]+conf_matrix[2,1])/nrow(spamdata_test)

cat("\n\nMissclassification Error with C=0.5:",missclas)
```