Lab3

Group8-lab3

12/14/2017

LAB 3 BLOCK 1: KERNEL METHODS AND NEURAL NETWORKS

1. KERNEL METHODS

In this assignment forecast consist of temperature from 4:00 to 24:00 in interval of 2 hours. We have implemeted the three kernels to calculate

1. station point(lat,lng) from point of interest(lat,lng)
2. date distance from point of interest date
3. hour distance

we have make this three kernels function by setting the smoothing coeffcient or width for each kernel which involve most of the point to consider

- Distacne kernel (h__distance = 1000000)
- Date kernel (h__date = 12)
- Time kernel (h__time = 7)

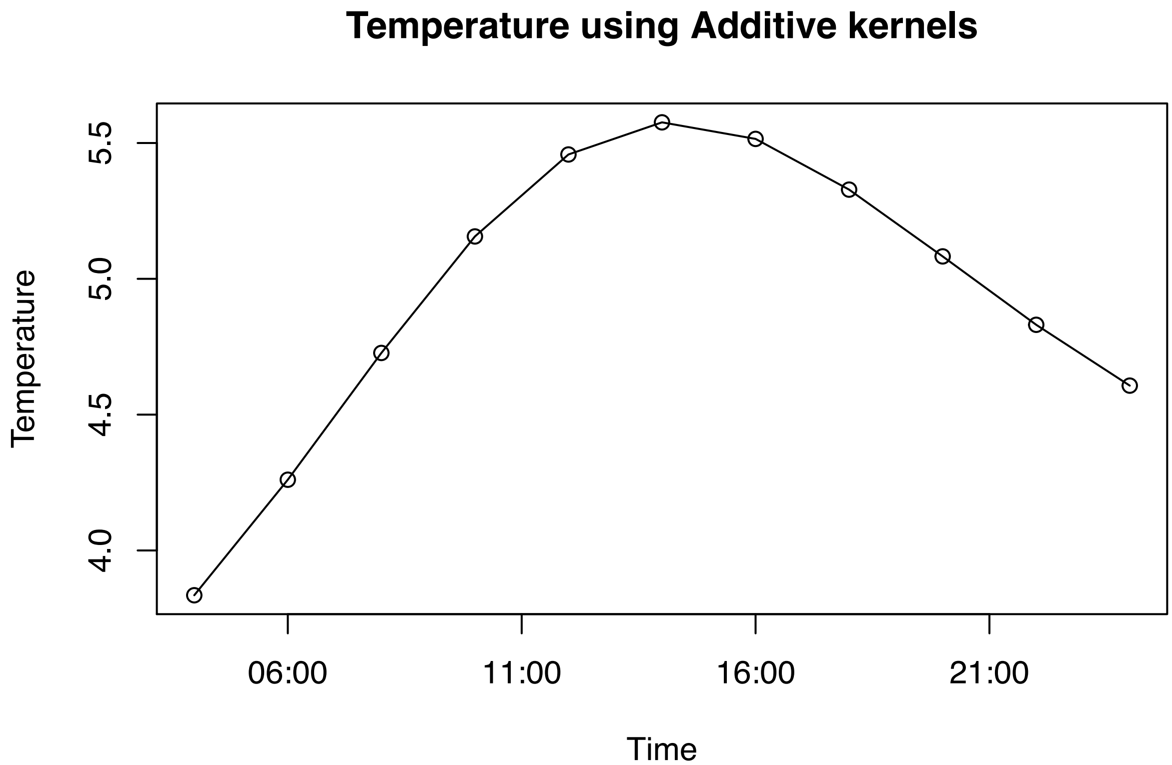
The all these three kernels were built on Gaussian kernel which is expressed as

$$kernel = exp^{-\frac{(x-\bar{x})}{h}}$$

```
#temperature values after added a sum of kernel weights
temp
```

```
##  [1] 3.835112 4.260383 4.727093 5.156020 5.457799 5.576042 5.515154
##  [8] 5.328433 5.082580 4.830791 4.606805
```

```
#plot of additive kernel
plot(x=times,y=temp, type = "o" ,
     main = "Temperature using Additive kernels" ,
     xlab = "Time" ,
     ylab = "Temperature")
```

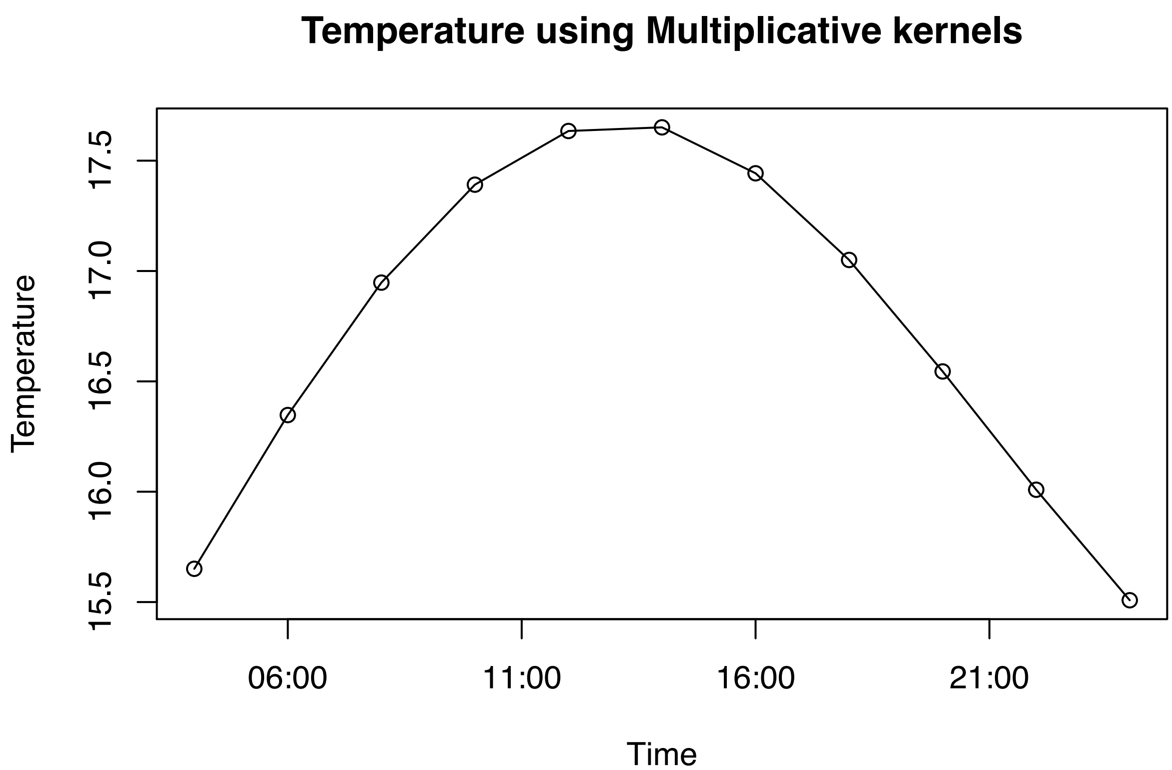


From plot we can see that X-axis has time and Y-axis is the additive temperature values and it can be observed that maximum temperature will be at time `15:00` and temperature is `5.8 degree` and minimum temperature is at some what `5:00` having a temperature `3.0 degree`

```
#temperature values after added a sum of kernel weights
temp2
```

```
##  [1] 15.65067 16.34742 16.94738 17.39124 17.63451 17.65100 17.44289
##  [8] 17.05014 16.54490 16.00891 15.50823
```

```
# plot of multiplicative kernels
#par(mar=c(3,3,3,3))
plot(x=times,y=temp2, type = "o",
     main = "Temperature using Multiplicative kernels" ,
     xlab = "Time" ,
     ylab = "Temperature")
```



From plot we can see the that X-axis has times and Y-axis is the Multiplicative temperature values and it can be observed that maximum temperature will be at time `15:00` and temperature is `17.7 degree` and minimum temperature is at some what `5:00` having a temperature `15.53 degree`

2. NEURAL NETWORKS

Most Appropriate Threshold

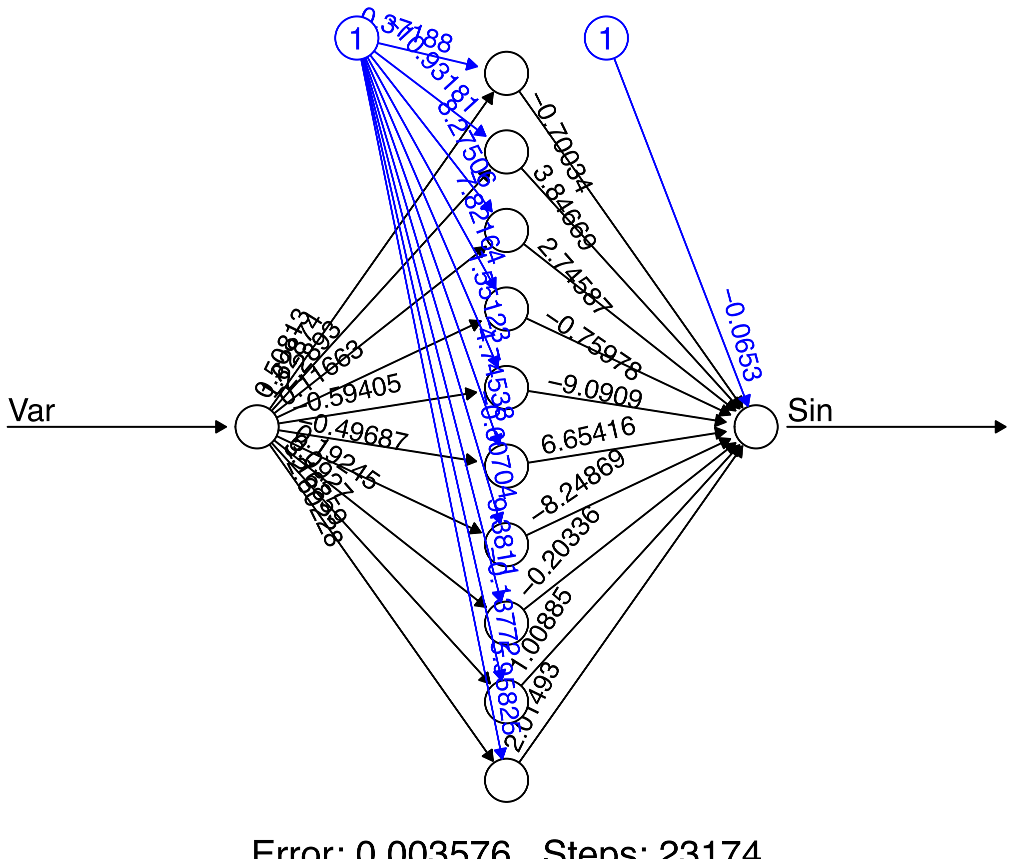
```
#threshold value for best
stoping_thresshold <- (which.min(va_MSE)) / 1000
stoping_thresshold
```

```
## [1] 0.004
```

the most appropriate value for thresshold is `0.0004`

```
nn <-  neuralnet(formula = Sin~Var ,data = tr,
                        hidden=c(10), threshold = stoping_thresshold,
                        startweights= winit)

#par(mar=c(2,2,2,2))
plot(nn, rep="best")
```

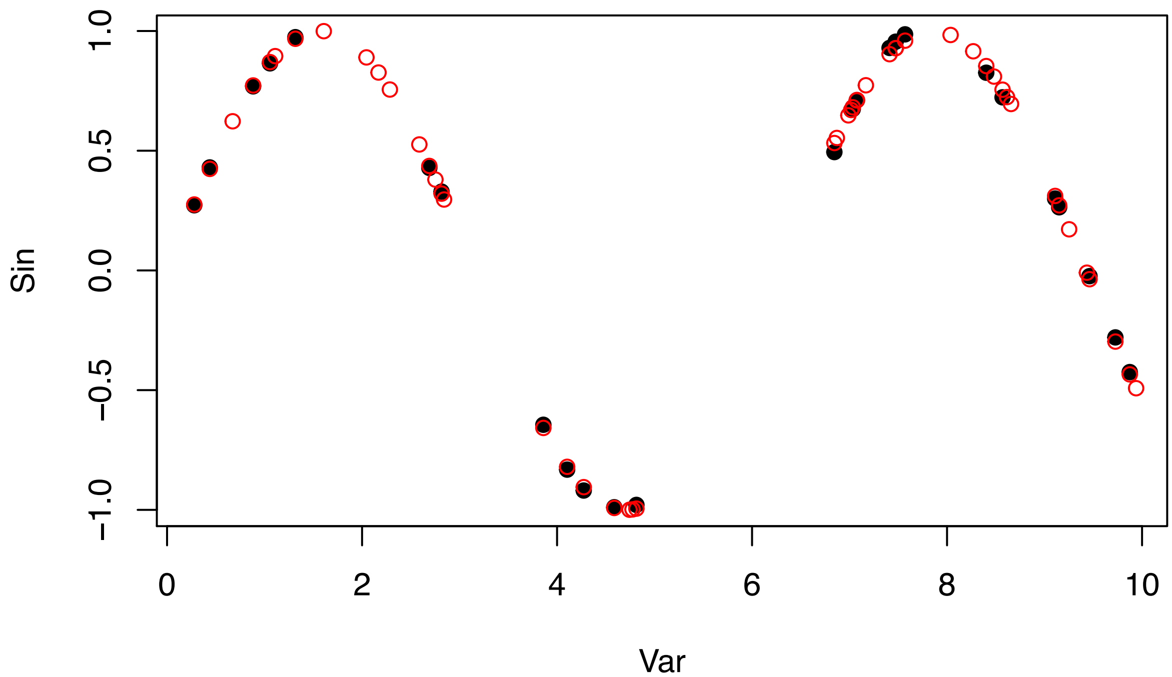


The final nerual network for trigonometric sine function learned at thresshold 4/1000 which is `0.0004` Plot implies that the, model fitted well ,there is not much deviation.

```
plot(prediction(nn)$rep1,pch=19, cex=1)
```

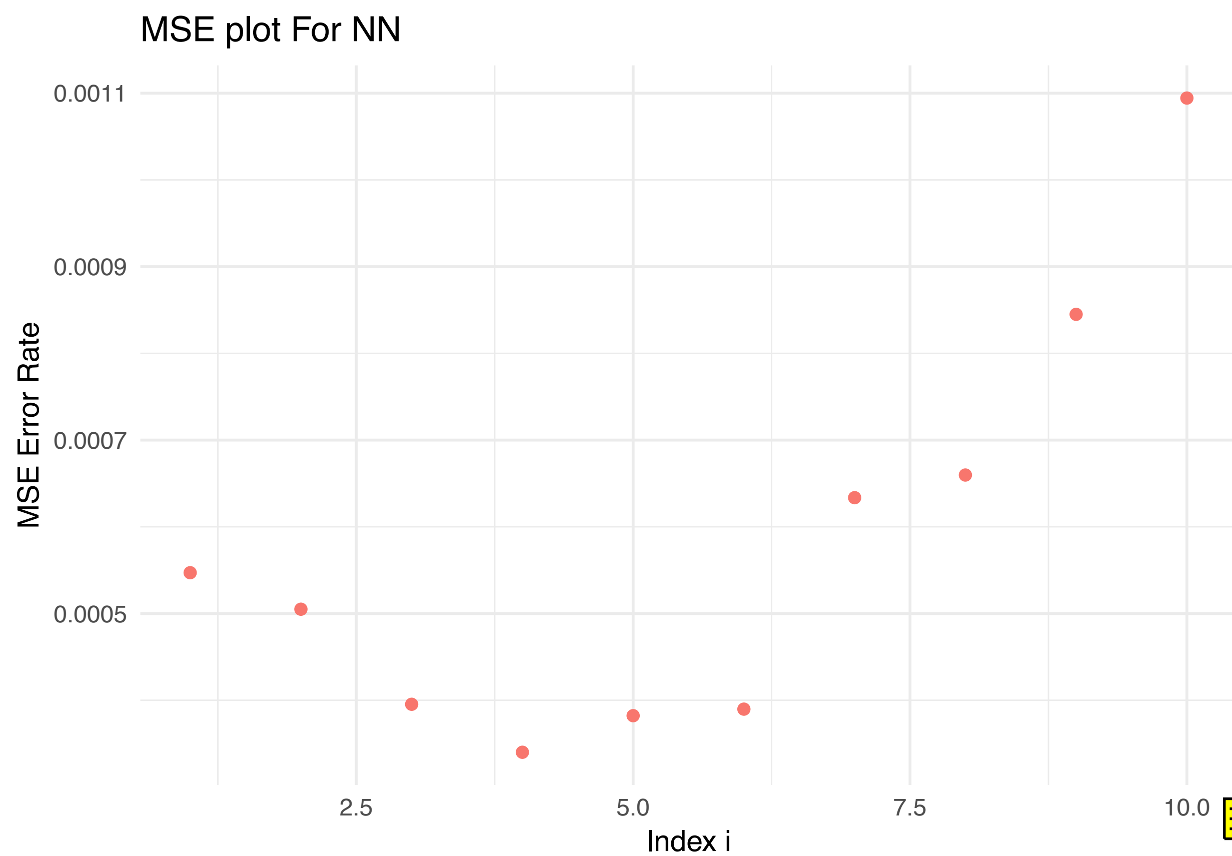
```
## Data Error:  0;
```

```
points(trva, col = "red")
```



This plot show the sinosedial sin waves with `Sin function` on Y-axis and `Var` on x-axis and the red dotted points curve show the original `sinosedial sin waves` and black dotted points curve show the predicted sin wave which is similar to the original one.

```
result <- data.frame(MSE=va_MSE, x=c(1:10))
ggplot(result, aes(x=x, y=MSE, color = "red")) +
  geom_point(shape = 16, size = 2, show.legend = FALSE) +
  theme_minimal() + ggtitle("MSE plot For NN") + xlab("Index i") +
  ylab("MSE Error Rate")
```



MSE plot show's that MSE error rate (Y-axis) on each $threshold = i/1000$ (X-axis) from plot it is clearly see that at index `i = 4 the error rate is minimum` so which give the optimal `Neural Network`

Appendix

1. KERNEL METHODS

```
library(neuralnet)
set.seed(1234567890)
library(geosphere)

stations <- read.csv("stations.csv",fileEncoding = "Latin1")
temps <- read.csv("temps50k.csv",fileEncoding = "Latin1")

st <- merge(stations,temps,by="station_number")
h_distance <- 1000000# These three values are up to the students
  h_date <- 12
  h_time <- 7
  a <- 59.8586 # The point to predict (up to the students)
  b <- 17.6253
date <- "2015-07-12" # The date to predict (up to the students)
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",
           "20:00:00","22:00:00","24:00:00")
temp <- vector(length=length(times))
```

```r
# Students' code here

#distance kernel
gaussion_distance <- function(db_point, point_intreset)
{
  #distance to the other point
  dist <- distHaversine(db_point, point_intreset)
  return (exp(-(dist / h_distance)^2)) #gaussian kernel
}

#gaussian date kernel
gaussian_date <- function(db_date, point_of_intreset_date)
{
  #date difference to other point
  diff_date <- as.numeric(difftime(point_of_intreset_date,db_date,unit = "days"))
  return (exp(-(diff_date / h_date)^2))
}

#gaussian time kernel
gaussian_hours <- function(db_time, point_of_intreset_date)
{
  #hours difference to other point
  diff_date <- as.numeric(difftime(point_of_intreset_date,db_time,unit = "hours"))
  return (exp(-(diff_date / h_time)^2))
}

point_intreset <- c(a,b) #point fo intreset
data_dist  = st[,c("longitude", "latitude")]

#calculate gaussian distance
gaussion_distacne_v<- gaussion_distance(data_dist, point_intreset)

#calculate gaussian date distance
gaussion_date_v <- gaussian_date(st$date,date)

time_conv <- data.frame(time=as.POSIXct(paste(Sys.Date(), st$time),
                                        format="%Y-%m-%d %H:%M:%S"))
times <- strptime( paste( Sys.Date(),times), "%Y-%m-%d %H:%M:%S")

temp2 <- vector(length=length(times))
# temp_type <- vector(length=length(times))
# temp2_type <- vector(length=length(times))

for (i in 1:length(times))
{
  gausian_hours <- gaussian_hours(time_conv$time, times[i])
  #sum of all kernels
  sum_of_k <- (gaussion_distacne_v + gaussion_date_v + gausian_hours)
  temp[i] <- sum((st$air_temperature * sum_of_k)) / sum(sum_of_k)
  # temp_type[i] <- "kernel_sum"

  #multiplication of kernels
  multiply_of_k <- (gaussion_distacne_v * gaussion_date_v * gausian_hours)
```

```r
  temp2[i] <- sum((st$air_temperature * multiply_of_k)) / sum(multiply_of_k)
  # temp2_type[i] <- "kernel_mul"
}

#plot of additive kernels
plot(x=times,y=temp, type = "o" ,
     main = "Temperature using Additive kernels" ,
     xlab = "Time" ,
     ylab = "Temperature")

#plot of multiplicative kernels
plot(x=times,y=temp2, type = "o",
     main = "Temperature using Multiplicative kernels" ,
     xlab = "Time" ,
     ylab = "Temperature")

# result <- data.frame(kernel=c(temp,temp2)  ,type=c(temp_type,temp2_type), x=times)
#
# library(ggplot2)
# ggplot(data=result, aes(x=x, y=kernel, colour=type)) + geom_line() + geom_point()
```

2. NEURAL NETWORKS

```r
library(neuralnet)
library(ggplot2)
set.seed(1234567890)
Var <- runif(50, 0, 10)
trva <- data.frame(Var, Sin=sin(Var)) #
tr <- trva[1:25,] # Training
va <- trva[26:50,] # Validation

# Random initialization of the weights in the interval [-1, 1]
va_MSE <- c()
tr_MSE <- c()

winit <- runif(31, -1, 1)
  for(i in 1:10) {
    nn <- neuralnet(formula = Sin~Var ,data = tr ,hidden = c(10),
                    threshold = i/1000, startweights= winit)
    va_predict <- compute(nn,va$Var)
    tr_predict <- compute(nn,tr$Var)
    va_MSE[i] <- sum( (va$Sin  - va_predict$net.result[,1])^2)/ nrow(tr)
    tr_MSE[i] <- sum( (va$Sin  - tr_predict$net.result[,1])^2)/ nrow(tr)

  }
  #comment
  #by visualizing the graph the thresshold value is 4/1000 which the stoping creteria
  #so NN model for such a thresshold is below

  #thresshold for stoping NN

  stoping_thresshold <- which.min(va_MSE)/1000
```

```r
  nn <- neuralnet(formula = Sin~Var ,data = tr ,hidden = c(10),
                  threshold = stoping_thresshold,
                  startweights= winit)
  plot(nn, rep="best" , main="Neural Network Plot")

  #another plot
  plot(prediction(nn)$rep1)
  points(trva, col = "red")

  #MSE plot
result <- data.frame(MSE=va_MSE, x=c(1:10))
ggplot(result, aes(x=x, y=MSE, color = "red")) +
  geom_point(shape = 16, size = 2, show.legend = FALSE) +
  theme_minimal() + ggtitle("MSE plot For NN") + xlab("Index i") +
  ylab("MSE Error Rate")
```