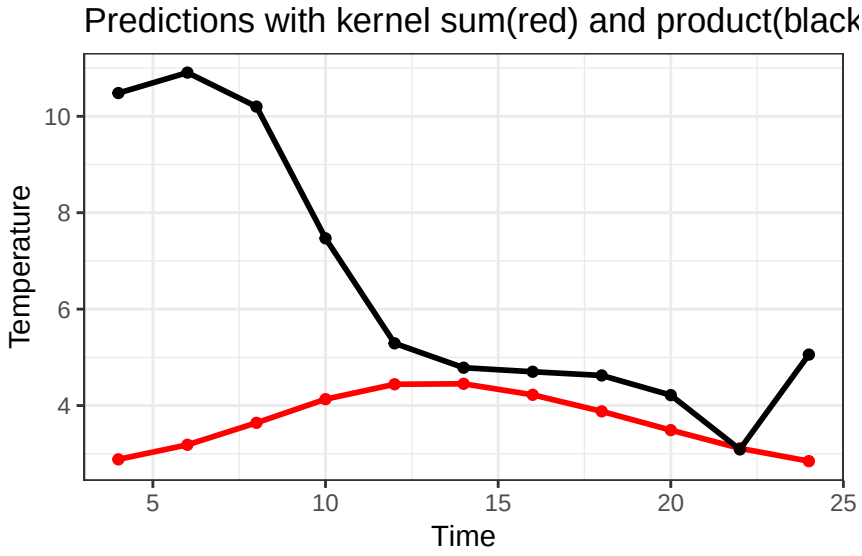
Lab3_block1

Andreas Stasinakis

December 13, 2018

Assignment 1

Task 1




```
##    Var1 Var2 Var3    error
## 9   0.1   40    3 56.37141
## 10  0.2   40    3 56.37141
## 11  0.3   40    3 56.37141
## 12  0.5   40    3 56.37141
## 13  0.1   50    3 56.89136
## 14  0.2   50    3 56.89136
## 15  0.3   50    3 56.89136
## 16  0.5   50    3 56.89136
## 5   0.1   30    3 57.98646
## 6   0.2   30    3 57.98646
```

Analysis

In this task we implement a kernal method in order to predict the hourly temperatures for a data and plane in Sweden. We use 3 different kernels and we combine them with 2 ways. Firstly we take the sum of them and after that the product. The most important thing in this procedure is the selection of the smoothing coefficient or width(h) for every different kernel. A sensible choice of smoothing coefficients would be the one below : i) $h_{distance} = 0.3$ ii) $h_{dates} = 40$ iii) $h_{time} = 3$

Those choices mean that we weight more the observations which are less than 300 kilometers away from the place of interest, less than 40 days after or before of the input's day and less than 3 hours difference from the hour interval selected. In general we should use cross validation, from 5 to 10 folds, in order to choose the optimal coefficients. In this case though, we have 50000 observations so it would be really difficult to

do it. In order to choose the optimal h we use *grid search* for a test dataset of 20 observations. We choose some different values for every smoothing coefficient. In all cases we choose some small values and some high values as boundaries. More specific, for the distance between a station and our place of interest we take a vector of 4 different values (0.1, 0.2 , 0.3, 0.5). The logic behind this choice is that we want to give more weight only for stations which their distances with the point of interest is 100 klm, 200 klm, 300 klm until 500klm (the distances are in meters so we divide by 1000 to convert them to kilometers. It does not make sense to pay attention in stations which their distance is more than 500. For the dates' smoothing coefficients we choose 5 different values (20,30,40,50,100). We are only interesting in observations which their date are not more than 100 days and that is the reason why our highest value of smoothing parameter is 100. We try to split the year in seasons so choosing a value more than 100 does not make that sense. For example, if we want to predict the weather for June and we choose $h = 150$ we will take all observations until November, but in this season the weather is totally different. Finally for the time smoothing parameter, we choose 3 different values (3,5,8). As we explain before, we are only interesting in period in a specific time period of the day, so the highest day interval is 8 hours. So we use the run a *grid search* algorithm for calculating the *MSE* for all h combinations in order to choose the optimal one.

The data frame of the *grid algorithm* we can verify the hypothesis above (only the head of it is printed). The minimum error for the test data is for $h_distance = 0.1 - 0.3$, so we can choose the last one. The estimation of h_date is 40 which also make sense as mentioned before. Finally h_time is estimated 3. As mentioned before, this test data set was really small but we use it only to take a general idea of our choice. The output is the expected one so the h's above seem to be the optimal.

We also make predictions using the product of the 3 kernels. The predictions seems really inaccurate fron 4 am until 12:00. After that we can say that they have similar values,but always higher, as the sum of the kernels. It can be also said that we can not make any conclusion about the distribution of this output or to find any pattern as in the sum which seems to be normally distributed. An explanasion for this may be the h's we select as optimal. As mentioned before those h was optimal for the sum of the kernels but not for the product. Also when we calculate the product all h's are correlated each other so an incorrect choice of one h can effect very much the output of the kernel. The best solution would be to run a *cross validation* or *grid search* to find the optimal smoothing coefficients as before.

Assignment 2

Task 2.1

```
library("kernlab")

data("spam")
data = spam

SVM_model = function(data){}

# import the data

n = dim(data)[1]
set.seed(12345)
id = sample(1:n, floor(n*0.5))
train = data[id,]
id1 = setdiff(1:n, id)
set.seed(12345)
id2 = sample(id1, floor(n*0.25))
valid = data[id2,]
id3 = setdiff(id1,id2)
test = data[id3,]
```

```r
# fit the SVM models with different C parameters

C = c(0.5,1,5)
models = list()

for ( i in C) {


  # fit the SVM models using Gaussian function with width 0.05
  models[[paste("SVM",i,sep = "_")]] = ksvm(type~., data = train, kernel = "rbfdot", kpar = list(sigma =

}

# make predictions for the 3 models
predict_SVM = list()

for (k in 1:3) {

  predict_SVM[[paste("SVM", k ,sep = "")]] =  predict(models[[k]], valid)
}



# calculate the misclassification rates for all 3 models

mis_rates = list()

for (i in 1:3) {
  mis_rates[[paste("SVM",i,sep = "")]] =  length(which(predict_SVM[[i]] != valid$type)) / nrow(valid)

}

# we use the misclassification rate to choose the best model

mis_rates
```

```
## $SVM1
## [1] 0.08608696
##
## $SVM2
## [1] 0.06956522
##
## $SVM3
## [1] 0.07478261
```

Task 2.2

```r
# From now one we will use only the best model using C = 1

best_model = models$SVM_1

# Estimate the Generalization error for the best model for unseen data(test data)
# predict for test data
```

```r
best_predicts = predict(best_model, newdata = test)

#calculate the misclassification rate
Gen_error = length(which(best_predicts != test$type)) / nrow(test)
```

Task 2.3

```
## $`Best model selected`
## 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 : 1007
##
## Objective Function Value : -467.7423
## Training error : 0.04
##
## $Generalization_Error
## [1] 0.08514335
```

Task 2.4

Purpose of C parameter

In this task we fit the data using 3 different SVM models with different C parameter. In general, we should do Cross - validation in order to find the best value of this C parameter. In this case though, we do not use CV but we fit the model using as $C = (0.1, 1, 5)$ and we conclude that the best model for these 3 values is the one using $C = 1$.

C parameter controls the trade off between a low training error and a low testing error that is the ability to generalize your classifier to unseen data. The value of parameter C defines the margin we want for the support vector. More specific, a high C values means small margin, as a result the hyperplane makes more accurate predictions for the training data *but* the misclassification rate for new unseen test data will probably be high (overfitting). On the contrary choosing a really small C means large margin, so the classifier does not care only for the close points. That may cause a lower test misclassification error but the classifier will be not good in the first place. In conclusion, the value of C parameter is really important for the efficiency of the SVM model and there is no value which garantie the best result. The optimal C value depends on the data and this is the reason why we should use Cross validation in order to choose the optimal one.