ISYE 6501 Intro Analytics Modeling – HW2

Question 3.1 Using the same data set (`credit_card_data.txt` or `credit_card_data-headers.txt`) as in Question 2.2, use the `ksvm` or `kknn` function to find a good classifier: (a) using cross-validation (do this for the k-nearest-neighbors model; SVM is optional); and (b) splitting the data into training, validation, and test data sets (pick either KNN or SVM; the other is optional).

In this step, I tried combinations of 10 different KNN kernel functions with 20 K values (# of neighbor considered,1-20).

By comparing the Mean Squared Error of **10-fold cross-validation,** I find the best model **with the smallest error 0.11.** The first figure shows how the average error changes across different kernels at K=10. (*Fig.1*)

```
Model Detail:
Call: train.kknn (formula = R1 ~ ., data = df,
                  kmax = 20, distance = 2, ks = 10,
                  kernel = knn_kernal, scale = TRUE)
Type of response variable: continuous
minimal mean absolute error: 0.1996336
Minimal mean squared error: 0.1113105
Best kernel: inv
Best k: 10
```

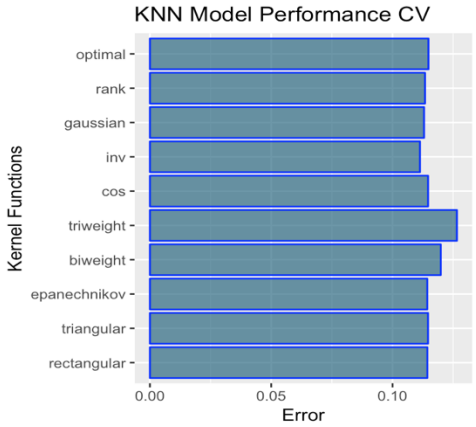

Fig.1

In the next step, I split the data into training, validation, and test data sets using the percentage of **70%, 15%,** and **15%.** After data training, we find **the smallest test error 0.15** using kernel function **Inv** and **optimal**. Then I fit the validation set on the Inv model and get **error 0.14.** So we can conclude that Inv model has the best performance is not because of the random effect of the test set, it is the real effect. (*Fig.2*)

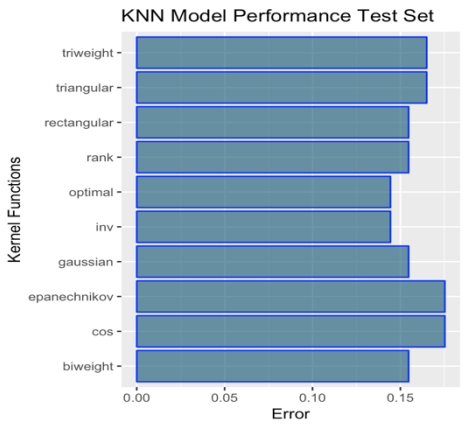

Fig.2

Question 4.1 Describe a situation or problem from your job, everyday life, current events, etc., for which a clustering model would be appropriate. List some (up to 5) predictors that you might use.

Clustering can be used to **separated different groups of people to make different preventive checks recommendations.** For example, cluster current eligible members to 5 categories (likelihood develop diabetes: 1: very likely-5: Very unlikely). And recommend related physic exams, e.g. blood sugar test to the people in group 1-2. Predictors: **age, sex, family history of diabetes, BMI, If smoking**.

Question 4.2 The *iris* data set `iris.txt` contains 150 data points, each with four predictor variables and one categorical response. The predictors are the width and length of the sepal and petal of flowers and the response is the type of flower. The data is available from the R library datasets and can be accessed with iris once the library is loaded. It is also available at the UCI Machine Learning Repository (https://archive.ics.uci.edu/ml/datasets/Iris). *The response values are only given to see how well a specific method performed and should not be used to build the model.* Use the R function `kmeans` to

cluster the points as well as possible. Report the best combination of predictors, your suggested value of k, and how well your best clustering predicts flower type.

The first step is to read in the dataset and do some simple data summary. From the code, I get to know this dataset contains **150 data points**, 4 **predictors** (all numeric), and **one response with 3 categories.** And they are evenly distributed (50 each) To find the best predictors, I tried 4 different algorithms, the scatterplots show how data points distribute, colors show prediction group and shape indicates the real group. The algorithm MacQueen gives the best performance with error **0.15.**

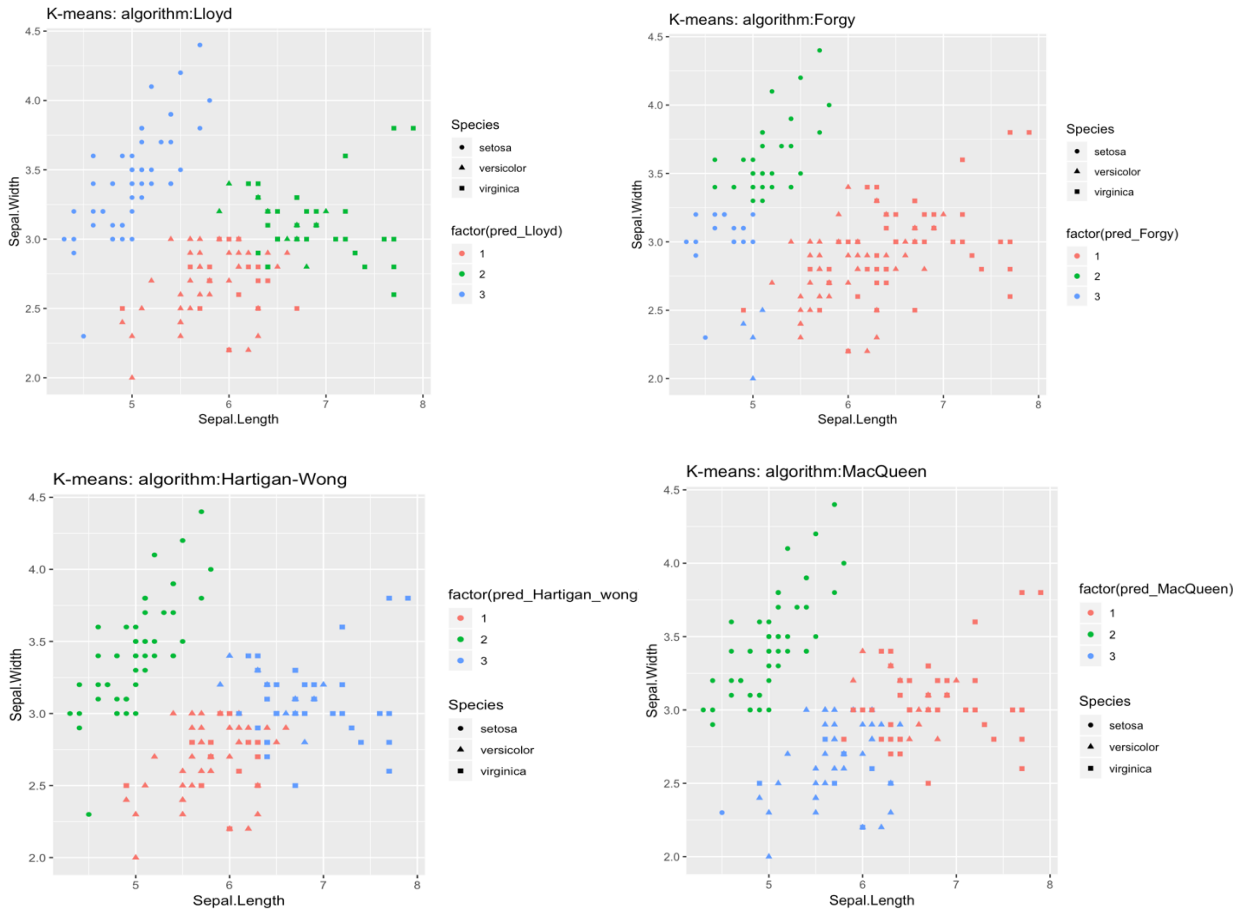

Fig.3

In the las step, I tried 20 centers values, to compare the clustering with 1 group to 20 groups. I used the value of the total within-cluster sum of squares as measurement. When K great than 10, y starts to decrease very slowly, so there is no big benefit to adding another cluster. (*Fig.4*)

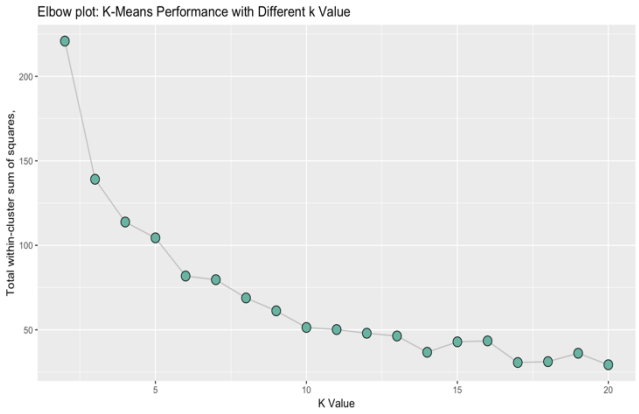

Fig. 4

The Code

```
SYE 6501 Intro Analytics Modeling - HW2
# IP credit_card_data-headers.txt
# IP iris.txt

# Loading data
df<-read.delim("data 3.1/credit_card_data-headers.txt", header = TRUE, sep =
"\t")

# Question 3.1

#### Library ####
library(kknn)
library(ggplot2)
library(dplyr)
#a-cross-validation#
knn_kernal=c("rectangular","triangular","epanechnikov","biweight","triweight"
,"cos", "inv", "gaussian", "rank", "optimal")
best_knn=train.kknn(R1~ ., df, kmax = 20, ks=10, distance=2, kernel = knn_ker
nal, scale = TRUE)
summary_table <-melt(best_knn$MEAN.SQU,id=c("row.names") ,na.rm = FALSE, valu
e.name = "MEAN_SQU")
ggplot(summary_table,aes(x=Var2, y=MEAN_SQU))+
  geom_bar(stat = "identity",color="blue", fill=rgb(0.1,0.4,0.5,0.7)) +
  coord_flip()+xlab("Kernel Functions") +ylab("Error") +
  labs(title ="KNN Model Performance CV")

#b#
set.seed(666)
g <- sample(1:3,size=nrow(df),replace=TRUE,prob=c(0.7,0.15,0.15))
train <- df[g==1,]
test <- df[g==2,]
validation <- df[g==3,]

summary_table2 <- data.frame(row.names=knn_kernal,k=knn_kernal)
for (x in knn_kernal){
  KNN<-kknn(R1~., train, test, distance =2,scale=TRUE, k=10,
            kernel = x)
  summary_table2[x,"tn"] = table(test$R1, round(KNN$fit))[1]
  summary_table2[x,"fn"] = table(test$R1, round(KNN$fit))[2]
  summary_table2[x,"fp"] = table(test$R1, round(KNN$fit))[3]
  summary_table2[x,"tp"] = table(test$R1, round(KNN$fit))[4]
  summary_table2[x,"error"]=(summary_table2[x,"fn"] +summary_table2[x,"fp"])/
nrow(test)
}
ggplot(summary_table2,aes(x=k, y=error))+
  geom_bar(stat = "identity",color="blue", fill=rgb(0.1,0.4,0.5,0.7)) +
  coord_flip()+xlab("Kernel Functions") +ylab("Error") +
```

```r
  labs(title ="KNN Model Performance Test Set")
error_test=min(summary_table2$error)
error_test

KNN_inv<-kknn(R1~., train, validation, distance =2,scale=TRUE, k=10,
              kernel = "inv")
error_val<-(table(validation$R1, round(KNN_inv$fit))[2]+table(validation$R1,
round(KNN_inv$fit))[3])/nrow(validation)
error_val

# Question 4.2

# Loading and examining data
df2<-read.table("iris.txt", header = TRUE)
#### Display Head Lines ####
head(df2,2)
#### Show attributes, number of row, and number of col####
attributes(df2)
ncol(df2)
nrow(df2)

#### Show the type, number of unique value, and summary of each variable ####
for (i in attributes(df2)$names ){
  print(paste0('Name:',i,'   Class:',class(df2[[i]]),'   nlevels:',length(uni
que(df2[[i]]))))
  print(summary(df2[[i]]))
}

df2_v<-scale(df2[,c(1:4)]) # Scaling the data
#clustring differenrt algorithm
for (a in c("Hartigan-Wong", "Lloyd", "Forgy","MacQueen")){
km<-kmeans(df2_v, centers=3, iter.max = 50, nstart = 1,
           algorithm=a, trace=FALSE)
assign(paste('pred', a, sep="_"),km$cluster)
}
pred_Hartigan_wong<-`pred_Hartigan-Wong`
summary_km <-cbind(df2,pred_Hartigan_wong,pred_Lloyd,pred_Forgy,pred_MacQueen
)

a<-"MacQueen"
ggplot(summary_km, aes(x=Sepal.Length, y=Sepal.Width, shape=Species, color=fa
ctor(pred_MacQueen))) +
             geom_point() +
             labs(title=paste0('K-means: algorithm:', a))

error<-(        nrow(summary_km[summary_km$Species == 'setosa' & summary_km$p
red_MacQueen !=2 ,])+
                 nrow(summary_km[summary_km$Species == 'versicolor' & summary
_km$pred_MacQueen !=3 ,])+
```

```
                  nrow(summary_km[summary_km$Species == 'virginica' & summary_
km$pred_MacQueen !=1 ,]))/nrow(summary_km)

#clustring differenrt k
kmean_withinss <- function(k) {
  km <- kmeans(df2_v, centers=k, iter.max = 50, nstart = 1,
               algorithm=a, trace=FALSE)
  return (km$tot.withinss)
}

max_k <-20
withiness <- sapply(2:max_k, kmean_withinss)
elbow <-data.frame(2:max_k, withiness)

  ggplot( elbow, aes(x=X2.max_k, y=withiness)) +
  geom_line( color="grey") +
  geom_point(shape=21, color="black", fill="#69b3a2", size=4) +
  labs(title ="Elbow plot: K-Means Performance with Different k Value")+ xlab
("K Value") +ylab("Total within-cluster sum of squares,")
```