MAFS 6010S - Final Project

03 June 2020

1. Introduction

This project aims to use Machine Learning Technique to build a prediction model to predict wheather a user will repeatly listen to a song within 1 month by using the KKBOX data. It is observed that interestingly the test data provided by KK box do not have the 'TARGET' input. Therefore, we will use the training data provided by KKBOX for both testing and training.

In particular, we choose Artificial Neural Network(ANN) as the method.

2. Data Selection and Data processing.

There are total 5 sets of data provided by KKBOX.

(1)members.csv; data that contains 1.1) city, 1.2) birthday, 1.3) gender, 1.4) registered_via, 1.5)registered_init_time, 1.6)expiration_date;

(2)song_extra_info.csv;2.1) name, 2.2) ISRC;

(3)song.csv;3.1) song_length, 3.2) genre_ids; 3.3)artist_name; 3.4)composer; 3.5) lyricist; 3.6)language

For (1), it is observed that in member.csv, the *bd*, *gender* are missing for many entries. therefore, we will omit these data as our input.

For (2), it is observed that in song_extra_info.csv, the *name*, *ISRC* are unique identifier of the song and should carry little information wheather a customer will repeatly listen to a song or not.

For (3), in song.csv, we only choose *language* as input. We would like to input the *genre_ids* , *song_length*, *artist_name*, *composer*, *lyricist* as input. However as we will be using class indicator that transfer categorical input to binary, if we includes these as input, the data matrix will be gigiantic and cannot be handled by R.

As the first step in the data prepration, we use the *join* function that is similiar to *vlookup* in Excel to join the data from *Train*, *members* and *songs*.

```
# we join the data and remove the data that is not used.
JointTrain<-join(train,members, by = "msno",type="left", match = "first")
JointTrain<-join(JointTrain,songs,by="song_id",type="left", match = "first")
# remove the data that is consider useless
JointTrain<-subset(JointTrain,select=-c(bd,gender,registration_init_time,expiration_date,song_length))
# Reorder the data
JointTrain1<-JointTrain[,c(6 ,1 ,2 ,3 ,4 ,5 ,7 ,8 ,9 ,10 ,11 ,12 ,13)]
```

Noticed the *nnet* function in R can process numerical data only. therefore, before inputing the the whole data as input, we need to use the *class.ind()* function to create some class indicator and change the categorical variable to binary input.

```
#Now, we apply ANN on training data set.
#all data are categorical. before we go into a ANN, we generate class indicator function. * we remove t
source_system_class <-class.ind(JointTrain1[,4])
source_screen_class<-class.ind(JointTrain1[,5])
source_type_class <-class.ind(JointTrain1[,6])
city_class<-class.ind(JointTrain1[,7])
registered_class <-class.ind(JointTrain1[,8])
language_class<-class.ind(JointTrain1[,13])

#Now we joint the data togather.
Real_TestData<-cbind(JointTrain1[,1:3],source_system_class,source_screen_class,source_type_class,city_cl

id<-sample(1:7377418,size=7327418) # get random row
TrainData1<-Real_TestData[-id,]
TestData1<-Real_TestData[+id,]
```

We would like to seperate testing data and training data into a 30:70 split. However, it is diffcult to input the data in ANN model with 70%*7.3mio data entry. Therefore, we limit the training data to smaller data size of 50,000 data entry.

3 Artificial Neural Network(ANN)

Now we write a Artificial Neural Network model. As ANN is very unstable in nature, we will use a iterative way to get the ANN model. in particular, we will fit the same data to ANN model 5 times and get the best one from each try.

```
ann<-function(x,y,size,maxit=500,linout=T,try=5){
  ann1<-nnet(y~.,data=x,size=size,maxit=maxit,linout=linout)
  v1<-ann1$value # save value for first trial

  for(i in 2:try){
    ann<-nnet(y~.,data=x,size=size,maxit=maxit,linout=linout)
    if(ann$value<v1){ #check if the current value is better
      v1<-ann$value #save the best value
      ann1<-ann #save result
    }
  }
  ann1 # return result
}
```

We fit the training data into ANN model with 1 hidden layer and 7,8,9,10 nodes.

Then found out the ANN10 has the smallest value and best for

The training misclassification rate is. 35.48% The testing misclassification rate is 38.56%

Both are less then 50%. the model improves from coin flipping.

```
#We have try ANN. but the memory size required is too large(~1TB) and my computer cannot provide such a
ann7$value
```

```
## [1] 11193.68
```

```
ann8$value
```

```
## [1] 11127.53
```

```
ann9$value
```

```
## [1] 11078.13
```

```
ann10$value
```

```
## [1] 11055.9
```

```
#ann10 has the lowest value, we take ann10
table(round(ann10$fit),TrainData1$target)
```

```
##
##         0     1
##   0 14750  7651
##   1 10024 17575
```

```
Predann10<-predict(ann10,TestData1[,4:83])
table(round(Predann10),TestData1$target)
```

```
##
##           0       1
##   0 2048432 1248404
##   1 1589556 2441026
```

4 Model Limitation

Despite the ANN model is good, there is some limitation. first of all, the class indicator transformation will make the data matrix very large and sparse. Meanwhile, we cannot

5 GitHub Link

https://github.com/hl88-stu/6010