Basic Cleaning of Health Index data

A.J.Nicholson

Introduction

This document records the basic cleaning of the Health Index data downloaded from http://global.sbs.ohio-state.edu/cd-contents/WH-Database.txt. The aim is to transform the data into a form which allows one to conduct exploratory analysis, with the greatest possible ease.

Data aquisition

On the website there are two versions of the data, WH-Database.txt and WH-Database.xls. For this I shall use the .txt version because this format is more easily read into r and the .xls version is incorrectly formatted.

```
url<-"http://global.sbs.ohio-state.edu/cd-contents/WH-Database.txt"
download.file(url,destfile="data/healthIndex.txt")
HIdata<-read.table("data/healthIndex.txt")
```

Step 1: Adding Column names

In the .txt version of the data the column names are missing. These are included in the other version of the data, so the information garnered from here can be added to the .txt data.

```
colnames<-c("Site","ID","Sex","Age","DentalAge","MinAge","MaxAge","DOB","Ancestry","SOC",
            "FDIAP","FLEN","HEIGT","FMIDA","FMIDM","HLEN","HCIR","LDI","LDC","LPI","LPC",
            "SUMTET","SUMPRE","SUMCAV","SUMSOK","SUMABS","CROB","PORHY","AUDEX","TIBINF",
            "SKELINF","DJSH","DJHK","DJCER","DJTHO","DJLUM","DJTMJ","DJWR","DJHAN","TRARM",
            "TRLEG","TRNAS","TRFAC","TRSKUL","TRHAN","TRWEAP")
names(HIdata)<-colnames
```

Step 2: Addressing Missing data

Currently the missing data is represented by periods (“.”). These need to be replaced by NA, the more standard way of representing missing data.

```
Fact2Num<- function(x){
  # this is a function to convert column classes from factors to numbers
  a<-as.character(x)
  a<-as.numeric(a)# will step will replace . with NA by coercion
  a
}

# Apply function to data
HIdata[,3:46]<-apply(HIdata[,3:46],2,FUN=Fact2Num)
```

Step 3:The Date of Birth column

```r
summary(HIdata$DOB)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##      NA      NA      NA     NaN      NA      NA   12520
```

This clearly shows that there is no data in this column so it can be removed.

```r
HIdata$DOB<-NULL
```

Step 4: Changing discrete data back to factors

A number of the columns contain categorical data. These are best represented as factors. In replacing periods with NAs all the data became of numerical class. This can be reversed for the discrete data as such:

```r
discrete<-c("Site","Sex","Ancestry","SOC",
            "LDI","LDC","LPI","LPC",
            "CROB","PORHY","AUDEX","TIBINF","SKELINF",
            "DJSH","DJHK","DJCER","DJTHO","DJLUM","DJTMJ","DJWR","DJHAN",
            "TRARM","TRLEG","TRNAS","TRFAC","TRSKUL","TRHAN","TRWEAP")
HIdata[,discrete]<-lapply(names(HIdata[,discrete]), function(x) as.factor(HIdata[,x]))
```

Step 5: Unique ID fields

To be of any use an ID field need to contain unique numbers, i.e. each unique record should have a different ID.

```r
ID<-HIdata$ID
uID<-length(unique(ID))
records<-length(ID)
```

Only using the ID column there are only 8609 unique IDs for 12520 recorded observations.

```r
ID<-HIdata[,c("Site","ID")]
uID<-length(unique(ID)[,1])
```

If you combine the ID and Site fields there are 12491 unique IDs for the 12520 observations.

```r
ID$duplicate<-duplicated(ID)# identify duplicates
dID<-ID[ID$duplicate==TRUE,] # extract Duplicate IDs
Duplicate<-HIdata[HIdata$ID %in% dID$ID & HIdata$Site %in% dID$Site,]#Extract all data for non unique IDs
```

There are 29 duplicated IDs used across 70 observations. Three of these observations are identical each with site = CHB and ID =1213Y. This is easily fixed by deleting the duplicates

```r
Duplicate<-Duplicate[!(Duplicate$Site=="CHB" & Duplicate$ID=="1213Y"),]
```

For the remaining the addition of a 2 to the end of the second instance of each ID could be sufficient to create unique Identifiers.

```
Duplicate$two<-duplicated(Duplicate[,c("Site","ID")])# find second instance
Duplicate$nID<-NA # new field
for(i in 1 : 67){
  Duplicate$nID[i]<-
    if(Duplicate$two[i]==TRUE){paste(Duplicate$ID[i],"2", sep="")
    }else{Duplicate$ID[i]}
}
summary(duplicated(Duplicate[,c("Site","nID")]))# test if now unique
```

```
##    Mode   FALSE    TRUE    NA's
## logical      66       1       0
```

This has made all but one pair unique. This in turn can be corrected by the addition of 3 onto the end of the ID of the second of these.

```
Duplicate$three<-duplicated(Duplicate[,c("Site","nID")])
Duplicate$nID[Duplicate$three==TRUE]<-paste(Duplicate$ID[i],"3", sep="")
```

Now that each unique record has a unique Identifier, the dataset can be reconstructed with these new IDs.

```
#Replace ID with nID
Duplicate$ID<-Duplicate$nID; Duplicate$nID<-NULL

#get back to original 45 fields
Duplicate$copy<-NULL; Duplicate$two<-NULL; Duplicate$three<-NULL

#add a single copy of the CHB 1213Y field
Duplicate<-rbind(Duplicate,unique(HIdata[HIdata$Site=="CHB" & HIdata$ID=="1213Y",]))

# get all other data
HIdata2<-HIdata[!(HIdata$ID %in% dID$ID & HIdata$Site %in% dID$Site),]

#combine the two
HIdata<-rbind(HIdata2,Duplicate)

#add a combined ID field to be used as reference in exploritory analysis
HIdata$cID<-paste(HIdata$Site,HIdata$ID,sep="_")
```

Next Step

This concludes the cleaning of the data.The next step is exploritory analysis which is is outlined in “Exploritory Analysis”.

```
write.table(HIdata,"data/HIdata.txt",row.names=FALSE)# saving the clean data.
cols<-lapply(HIdata,class);cols<-rbind(cols)# stores the classes of the columns
write.table(cols,"data/cols.txt",row.names=FALSE)
```