install.packages(‘igraph’)

install.packages(‘dplyr’)

install.package(‘pheatmap’)

install.package(‘bnlearn’)

library(igraph)

me\_16s\_top100<-read.table('root\_16s\_top.txt',sep = '\t',header = T)

me\_ITS\_top100<-read.table('root\_ITS\_top.txt',sep = '\t',header = T)

me\_16s\_1<-me\_16s\_top100[,-c(194:198)]

me\_ITS\_1<-me\_ITS\_top100[,-c(194:198)]

me\_16s\_2<-me\_16s\_1[,-c(1:3)]

me\_ITS\_2<-me\_ITS\_1[,-c(1:3)]

me\_16s\_3<-t(me\_16s\_2)

me\_ITS\_3<-t(me\_ITS\_2)

me\_16s\_4<-as.numeric(me\_16s\_3)

me\_ITS\_4<-as.numeric(me\_ITS\_3)

me\_16s\_5<-matrix(me\_16s\_4,nrow=190,ncol = 100)

me\_ITS\_5<-matrix(me\_ITS\_4,nrow = 190,ncol = 100)

root\_top<-cbind(me\_16s\_5,me\_ITS\_5)

root\_top100<-log10(root\_top+1)

#mutualism

y\_mutualism\_1<-matrix()

for(n in 1:179){

for (i in 1:199){

for (j in seq(i+1,200)){

if(root\_top100[n,i] > root\_top100[n, j])

{x1 <-((root\_top100[n,i]\*root\_top100[n, j])/(root\_top100[n,i]-root\_top100[n, j]))}

else

{x1 <-((root\_top100[n,i]\*root\_top100[n, j])/(root\_top100[n,j]-root\_top100[n, i]))}

y\_mutualism\_1 <- rbind(y\_mutualism\_1,x1)

}

}

}

#antagonism

y\_antagonism\_1<-matrix()

for(n in 1:179){

for (i in 1:199){

for (j in seq(i+1,200)){

if(root\_top100[n,i] > root\_top100[n, j])

{x2 <-(1/((root\_top100[n,i]\*root\_top100[n, j])\*(root\_top100[n,i]-root\_top100[n, j])))}

else

{x2 <-(1/((root\_top100[n,i]\*root\_top100[n, j])\*(root\_top100[n,j]-root\_top100[n, i])))}

y\_antagonism\_1 <- rbind(y\_antagonism\_1,x2)

}

}

}

#aggression

y\_aggresion\_1<-matrix()

for(n in 1:179){

for (i in 1:199){

for (j in seq(i+1,200)){

if(root\_top100[n,i] > root\_top100[n, j])

{x3 <-((root\_top100[n,i]/root\_top100[n, j]))}

else

{x3 <-((root\_top100[n,j]/root\_top100[n, i]))}

y\_aggresion\_1 <- rbind(y\_aggresion\_1,x3)

}

}

}

#altruism

y\_altruism\_1<-matrix()

for(n in 1:179){

for (i in 1:199){

for (j in seq(i+1,200)){

if(root\_top100[n,i] > root\_top100[n, j])

{x4 <-(1-(root\_top100[n,j]/root\_top100[n, i]))}

else

{x4 <-(1-(root\_top100[n,i]/root\_top100[n, j]))}

y\_altruism\_1 <- rbind(y\_altruism\_1,x4)

}

}

}

y\_mutualism1<-y\_mutualism[-1,]

y\_mutualism2<-matrix(y\_mutualism1,ncol=179)

y\_mutualism\_t<-t(y\_mutualism2)

y\_mutualism\_t[is.infinite(y\_mutualism\_t)]<-0

y\_mutualism\_t[is.na(y\_mutualism\_t)]<-0

y\_mutualism\_t[is.nan(y\_mutualism\_t)]<-0

#Using the same code to get y\_antagonism\_t, y\_aggression\_t and y\_altruism\_t

#[0,1]

m1 <-matrix()

for(i in 1:179){

for (j in seq(ncol(y\_mutualism\_t))){

x<-(y\_mutualism\_t[i,j]-min(y\_mutualism\_t[i,]))/(max(y\_mutualism\_t[i,]-min(y\_mutualism\_t[i,])))

m1 <- rbind(m1,x)

}

}

m1<-m1[-1,]

m1\_<-matrix(m1,nrow = 19900)

mutualism\_0\_1<-t(m1\_)

#Using the same code to get antagonism\_0\_1, aggression\_0\_1 and mutualism\_0\_1

for(i in 1:nrow(mutualism\_0\_1)){

for(j in 1:ncol(mutualism\_0\_1)){

if(mutualism\_0\_1[i,j]<0.95){

mutualism\_0\_1[i,j]<-0

}

}

}

for(i in 1:nrow(antagonism\_0\_1)){

for(j in 1:ncol(antagonism\_0\_1)){

if(antagonism\_0\_1[i,j]<0.95){

antagonism\_0\_1[i,j]<-0

}

}

}

for(i in 1:nrow(aggression\_0\_1)){

for(j in 1:ncol(aggression\_0\_1)){

if(aggression\_0\_1[i,j]<0.95){

aggression\_0\_1[i,j]<-0

}

}

}

for(i in 1:nrow(altruism\_0\_1)){

for(j in 1:ncol(altruism\_0\_1)){

if(altruism\_0\_1[i,j]<0.99){

altruism\_0\_1[i,j]<-0

}

}

}

data1<-matrix(apply(mutualism\_0\_1,2,sum),ncol = 1)

data2<-matrix(apply(antagoonism\_0\_1,2,sum),ncol = 1)

data3<-matrix(apply(aggression\_0\_1,2,sum),ncol = 1)

data4<-matrix(apply(altruism\_0\_1,2,sum),ncol = 1)

A<-list()

for(i in 1:200){x<-rep(i,200-i)

A<-c(A,x)}

A<-as.numeric(A)

B<-list()

for(i in 2:200){x<-seq(i,200)

B<-c(B,x)}

B<-as.numeric(B)

o<-cbind(A,B)

o\_1<-cbind(o,data1)

o\_2<-cbind(o,data2)

o\_3<-cbind(o,data3)

o\_4<-cbind(o,data4)

mutualism\_<-o\_1[-which(o\_1[,3]==0),]

antagonism\_<-o\_1[-which(o\_2[,3]==0),]

aggression\_<-o\_1[-which(o\_3[,3]==0),]

altruism\_<-o\_1[-which(o\_4[,3]==0),]

net\_mutualism<-graph\_from\_data\_fram(mutualism\_,directed=F)

net\_antagonism<-graph\_from\_data\_fram(antagonism\_,directed=F)

net\_aggression<-graph\_from\_data\_fram(aggression\_,directed=F)

net\_altruism<-graph\_from\_data\_fram(altruism\_,directed=F)

plot(net\_mutualism)

plot(net\_antagonism)

plot(net\_aggression)

plot(net\_altruism)

#heatmap

library(pheatmap)

#mutualism

M2<-matrix()

for(i in 1:179){

mutu\_2<-cbind(o,t(mutualism\_0\_1[i,]))

mutu\_2delete<-as.factor(mutu\_2[-which(mutu\_2[,3]==0),])

m<-matrix(mutu\_2delete,ncol = 3)

net\_mutu<-graph\_from\_edgelist(m[,1:2],directed = T)

con1<-mean(degree(net\_mutu))

n<-mean(closeness(net\_mutu))

B2<-mean(betweenness(net\_mutu))

E2<-mean(eccentricity(net\_mutu))

G2<-mean(eigen\_centrality(net\_mutu)$vector)

P2<-mean(page.rank(net\_mutu)$vector)

M2<-c(M2,con1,n,B2,E2,G2,P2)

}

M2<-as.matrix(M2)

M2<-M2[-1,]

M2\_<-matrix(M2,nrow=6)

M2\_2<-t(M2\_)

colnames(M2\_2)<-c('con','C(u)','B(u)','E(u)','G(u)','P(u)')

write.csv(M2\_2,file = 'mutualism\_heatmap.csv')

mutualism<-read.csv('mutualism\_heatmap.csv',header = T,sep = ',')

mutualism<-as.matrix(mutualism)

mutualism<-mutualism[,-1]

colnames(mutualism)<-c('Con','C(u)','B(u)','E(u)','G(u)','P(u)')

M<-matrix()

for(i in 1:6){

for(j in 1:179){

m<-(mutualism[j,i]-min(mutualism[,i]))/(max(mutualism[,i])-min(mutualism[,i]))

M<-rbind(M,m)

}}

M<-M[-1,]

M<-matrix(M,nrow = 179)

colnames(M)<-c('Con','C(u)','B(u)','E(u)','G(u)','P(u)')

write.csv(M,file = 'mutualism\_heatmap\_0\_1.csv')

mutualism\_heatmap\_0\_1<-read.csv('mutualism\_heatmap\_0\_1.csv',header = T,sep=',')

row.names(mutualism)<-seq(1,179)

pheatmap(mutualism,scale= 'none',color = colorRampPalette(colors = c('blue','white','red'))(100),

show\_rownames = T,fontsize = 10,fontsize\_row = 4,cluster\_cols=F)

#Using the same code to get the heatmap of the other three relationships

#GWAS

#mutualism

mutualism<-read.csv('mutualism\_heatmap\_0\_1.csv',header = T)

mutualism<-mutualism[,-1]

SNP179\_<-read.csv('SNP179\_0\_1.csv',header=T,sep=',')

SNP179\_<-SNP179\_[,-1]

mutualism<-as.matrix(mutualism)

#Con

b<-c("Estimate","Std. Error","t value","Pr(>|t|)" )

M<-matrix(ncol = 4)

colnames(M)<-b

for (i in 1:201848){

SNP179\_P<-as.numeric(SNP179\_[i,])

lm\_1<-as.data.frame(cbind(SNP179\_P,mutualism[,1]))

colnames(lm\_1)<-c('b1','Con')

mode1<-lm(Con ~ b1,data = lm\_1)

a1<-as.data.frame(summary(mode1)$coef)

m1<-rbind(m1,a1[2,])

}

M\_<-M[-1,]

M\_<-as.matrix(M\_)

data<-read.csv('chr\_201848positions.csv',header = T)

data<-data[,-1]

M\_1<-cbind(data,M\_)

write.csv(M\_1,file = 'Right\_mutualism\_Con\_p.csv')

M\_1[is.na(M\_1)]<-1

P\_<-cbind(M\_1[,1:2],M\_1[,10])

colnames(P\_)<-c('CHR','BP','P')

manhattan(P\_,main='Con\_mutualism',col = c('blue4','orange3'))

write.csv(P\_,file = 'Manhatton\_Con\_mutualism.csv')

#Using the same code to get the manhatton of the other properties

#Bayesian network

library(dplyr)

library(bnlearn)

SNP179\_<-read.csv('SNP179\_0\_1.csv',header=T,sep=',')

SNP\_matrix<-as.matrix(SNP179\_)

significant\_snp<-read.table('significant\_snp\_fuben.txt',header = T,sep = '\t')

M<-as.matrix(significant\_snp)

M1<-matrix(as.numeric(M[,2]),ncol = 1)

#mutualism

mutualism<-read.csv('mutualism\_heatmap\_0\_1.csv',header = T)

mutualism<-mutualism[,-1]

mutualism<-as.matrix(mutualism)

# the significant SNP of Con\_mutualism

b\_con\_mutu<-matrix(ncol = 180)

for(i in 1:8){

a<-SNP\_matrix[which(SNP\_matrix[,1]== M1[i,1]),]

b\_con\_mutu<-rbind(b\_con\_mutu,a)

}

b\_con\_mutu<-b\_con\_mutu[-1,]

b\_con\_mutu<-t(b\_con\_mutu)

B\_con\_mutu<-b\_con\_mutu[-1,]

colnames(B\_con\_mutu)<-c(1:8)

#Combining genotypes and phenotypes

con\_mutu\_combine<-data.frame(as.numeric(mutualism[,1]),B\_con\_mutu)

colnames(con\_mutu\_combine)<-c('phenotypes',1:8)

con\_mutu\_combine\_1<-as.matrix(con\_mutu\_combine)

con\_mutu\_combine\_2<-con\_mutu\_combine\_1[,-1]

#Phenotypes mean

B1\_con\_mutu<-matrix(ncol = 2,nrow = 2)

for(i in 2:9){

count\_phenomeans\_diffgenetype<-con\_mutu\_combine %>% select(phenotypes,con\_mutu\_combine[,i]) %>%

group\_by(con\_mutu\_combine[,i]) %>% summarise\_at(vars(phenotypes),mean)

count\_mean\_con\_mutu<-as.matrix(count\_phenomeans\_diffgenetype)

B1\_con\_mutu<-cbind(B1\_con\_mutu,count\_mean\_con\_mutu)

}

B1\_con\_mutu<-B1\_con\_mutu[,-c(1:2)]

A\_con\_mutu<-matrix()

for(i in 1:8){

for(j in 1:179){

n<-2\*i-1

a1<-B1\_con\_mutu[which(con\_mutu\_combine\_2[j,i]== B1\_con\_mutu[,n]),n+1]

A\_con\_mutu<-rbind(A\_con\_mutu,a1)

}

}

A\_con\_mutu<-A\_con\_mutu[-1,]

A1\_con\_mutu<-matrix(A\_con\_mutu,nrow = 179)

colnames(A1\_con\_mutu)<-c(1:8)

library(bnlearn)

bn\_con\_mutu<-hc(as.data.frame(A1\_con\_mutu))

bn1\_con\_mutu<-amat(bn\_con\_mutu)

library(igraph)

net\_con\_mutualism<-graph.adjacency(bn1\_con\_mutu,mode = 'directed')

V(net\_con\_mutualism)$color<-'#1c86EE'

V(net\_con\_mutualism)[degree(net\_con\_mutualism)>2]$color<-'#228B22'

plot(net\_con\_mutualism,main='Con\_mutualism',layout=layout\_in\_circle,label.color='black',edge.color='#483D8B',edge.arrow.size=1,vertex.size=13,vertex.label.cex=1.2)

#Using the same code to get the Bayesian network of the other properties