Data Mining
Home work 05
Q-Q-plots and frequent itemsets

Aqeel Labash
Lecturer: Jaak Vilo

6 March 2016

First Question

Use the data of child height/weight and study them using qq-plots in a specific age and gender at a time.
Firstly I removed unneeded data and removed the negative values by this code :

```
rm(list=ls())
setwd('/home/aqeel/Study/DM/HW05')
ncmp = read.csv('ncmp_1415_final_non_disclosive.csv',header = TRUE)
ncmp$ncmppseudosystemid<-NULL
ncmp$heightzscore<-NULL
ncmp$suppress_imd<-NULL
ncmp$heightpscore<-NULL
ncmp$weightzscore<-NULL
ncmp$weightpscore<-NULL
ncmp$bmizscore<-NULL
ncmp$bmipscore<-NULL
ncmp$suppress_table<-NULL
ncmp$suppress_record_low<-NULL
ncmp$schooltier1localauthority<-NULL
ncmp$suppress_record_high<-NULL
ncmp$suppress_record_high<-NULL
ncmp$pupilschooldistancebanded<-NULL
ncmp$schooltier2localauthority<-NULL
ncmp$schoolgovernmentofficeregion<-NULL
ncmp <- ncmp[ncmp$height >0,]
datasample<-ncmp[sample(nrow(ncmp),5000),]
```

After that I tried to get the age with most kids in it by running this code :

```
test <-table(ncmp$ageinmonths)
test[test==max(test)]
```

Which lead to age 60.6 with 7346 kid.I used the following code to separate male,females and plot height vs weight under age 60.6

```
male<- ncmp[ncmp$ageinmonths==60.6&
ncmp$genderdescription=="Male",]
female<-ncmp[ncmp$ageinmonths==60.6&
ncmp$genderdescription=="Female",]

png ('male.png',width = 1200,height = 830)
qqplot(male$height,male$weight)
dev.off()
png ('female.png',width = 1200,height = 830)
qqplot(female$height,female$weight)
dev.off()
```

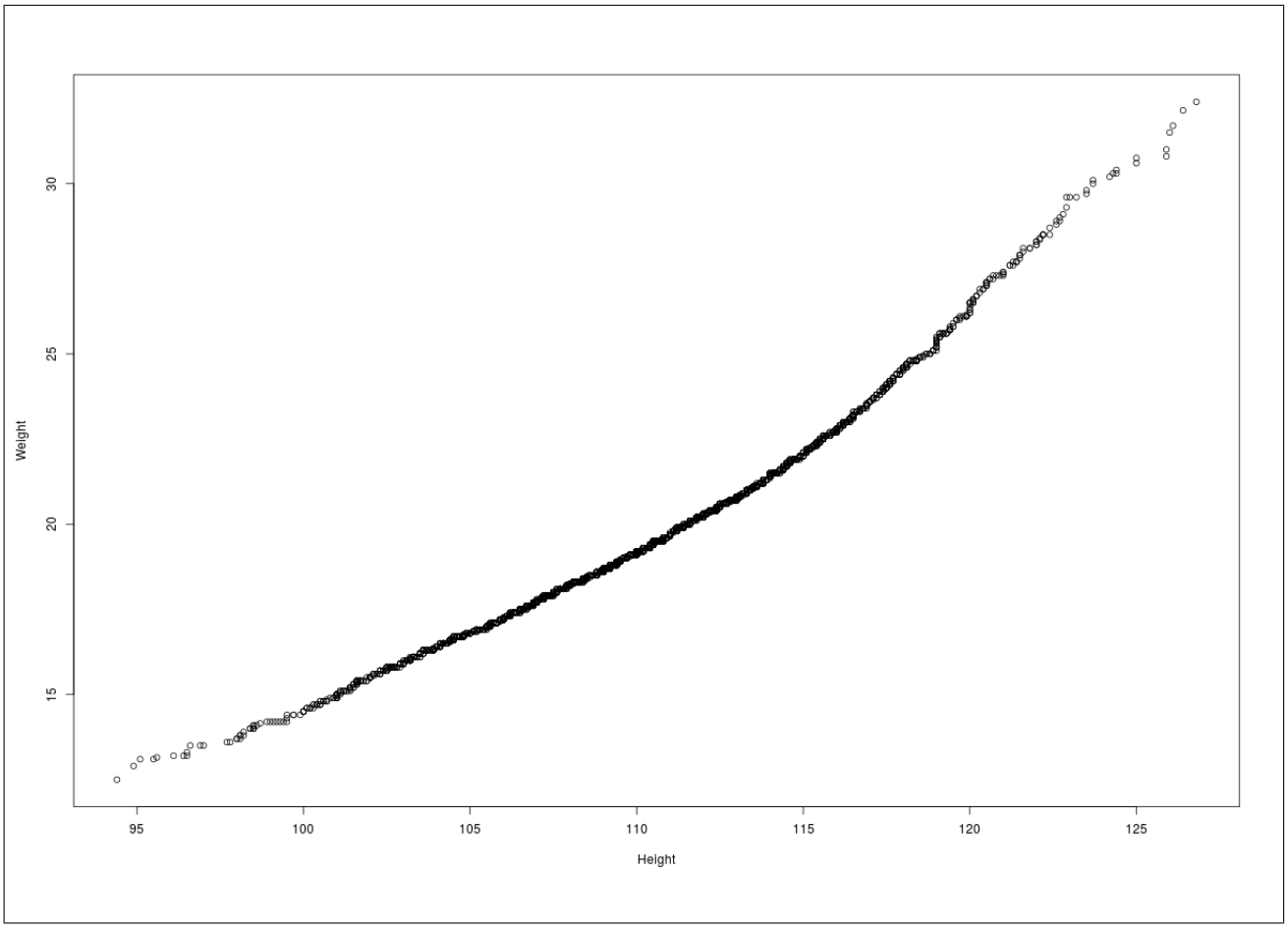


Figure 1: Weight vs height for male kids at age 60.6

From figure 1 can notice that height and weight are highly correlated.

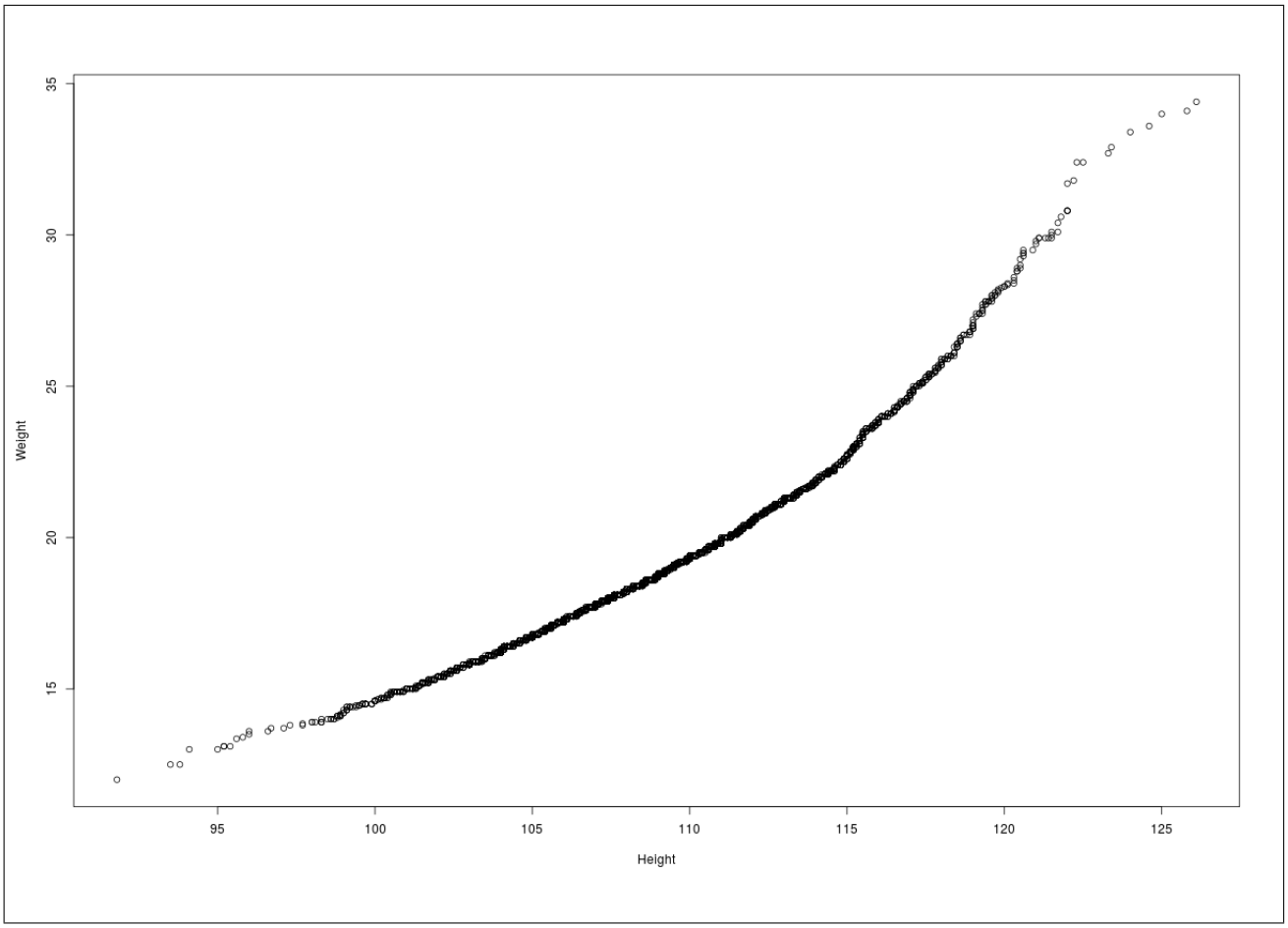


Figure 2: Weight vs height for male kids at age 60.6

In figure 2 we can notice that they are also correlated but if we compare figure 1 with figure 2 we can notice that the weight growth with height slightly higher.

Compare heights of underweight and very overweight children

To do the comparison I used the following code :

```
underoverweights<- ncmp[ncmp$bmipopulationcategory =='very overweight' | ncmp$
    bmipopulationcategory=='underweight' ,]
png('undervsover.png',width=1200,height = 830)
ggplot(data = underoverweights, mapping = aes(x=bmipopulationcategory,y=height))+geom_point()
dev.off()
```

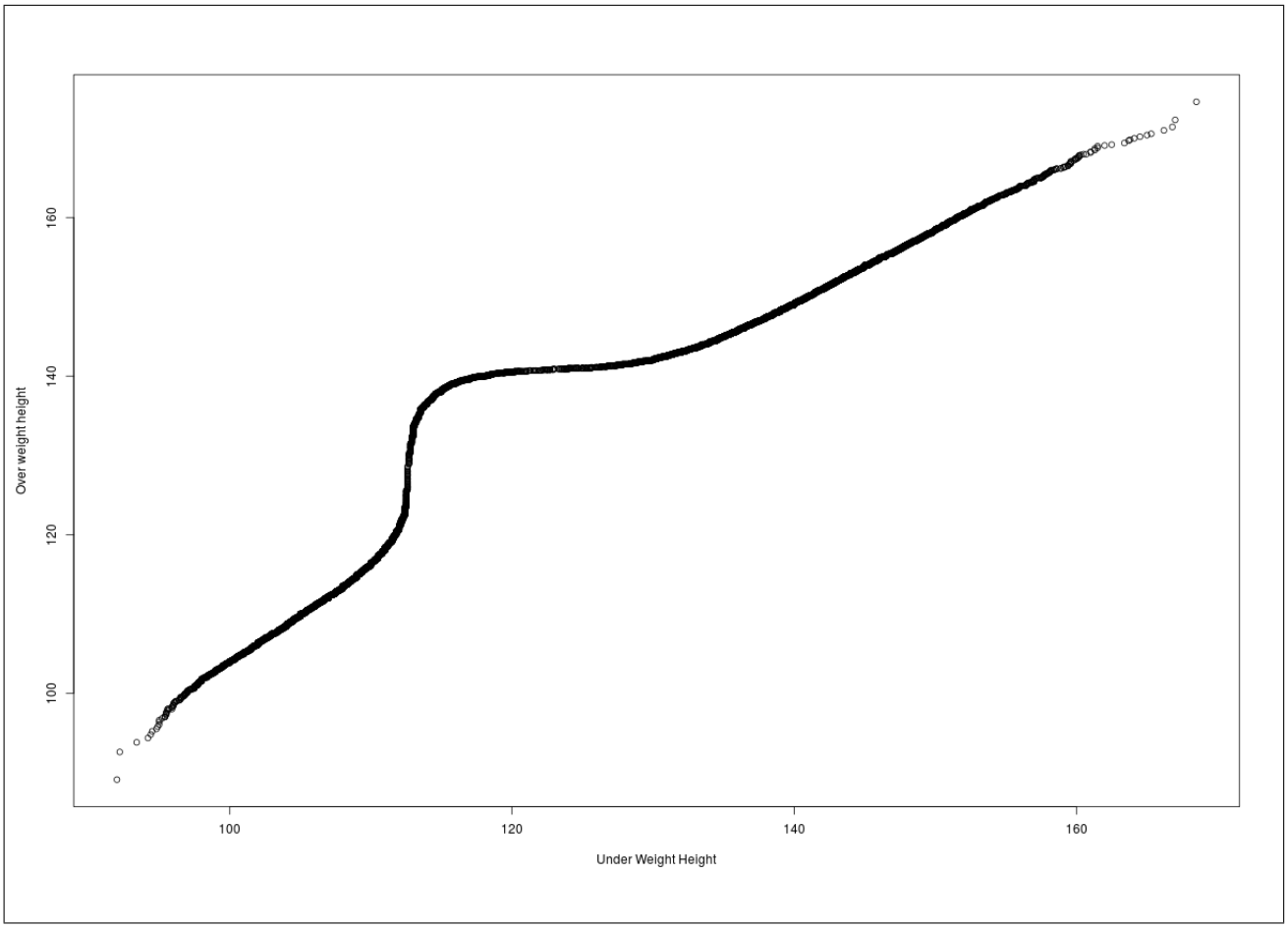


Figure 3: Comparison under weight vs very over weight height

From figure 3 we can see that over weight kids and under weight kids have almost same distribution but I would say that there is high percentage of under weight kids with heights around 110.The following code is what I used to accomplish it:

```
underweight<- ncmp[ncmp$bmipopulationcategory =='very overweight' ,]
overweight<-ncmp[ncmp$bmipopulationcategory=='underweight',]
png('undervsover.png',width=1200,height = 830)
qqplot(underweight$height,overweight$height,xlab = 'Under Weight Height',ylab = 'Over weight
    height')
dev.off()
```

Compare one of the attributes (height, weight, BMI) between boys and girls (select either younger or older age group) to each other.

For this task I selected the younger age group. and used the following code :

```
agelimits<-c(49.1,70.0,120.8,141.4)
maleage1 <-ncmp[ncmp$ageinmonths>agelimits[1] & ncmp$ageinmonths<agelimits[2]&
ncmp$genderdescription=="Male",]

femaleage2 <-ncmp[ncmp$ageinmonths>agelimits[1] & ncmp$ageinmonths<agelimits[2]&
ncmp$genderdescription=="Female",]
png('malevsfemale.png',width=1200,height = 830)
qqplot(maleage1$height,femaleage2$height)
dev.off()
```

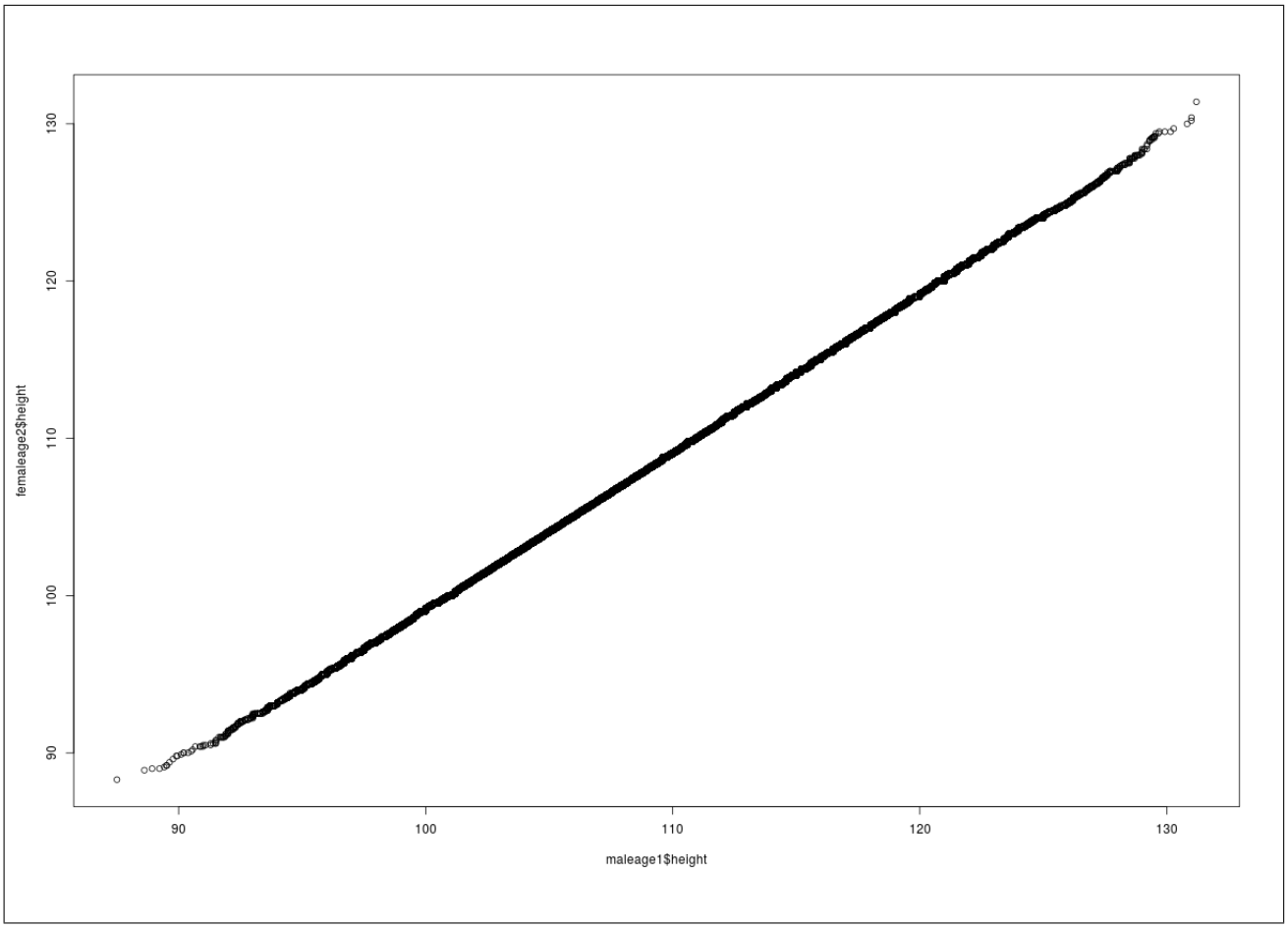


Figure 4: Q-Q plot for heights male vs female

From previous plot we can see that it's almost linear plot which mean at this age almost female and male have the same height.In other worlds sex doesn't affect the height at this age.

Second Question

For this question I used the groups I created for the first question with same age 60.6 and used the following code for the plots:

```
png('malebmi.png',width = 1200,height = 830)
qqplot(male$height,male$bmi)
dev.off()
png('femalebmi.png',width = 1200,height = 830)
qqplot(female$height,female$bmi)
dev.off()
```

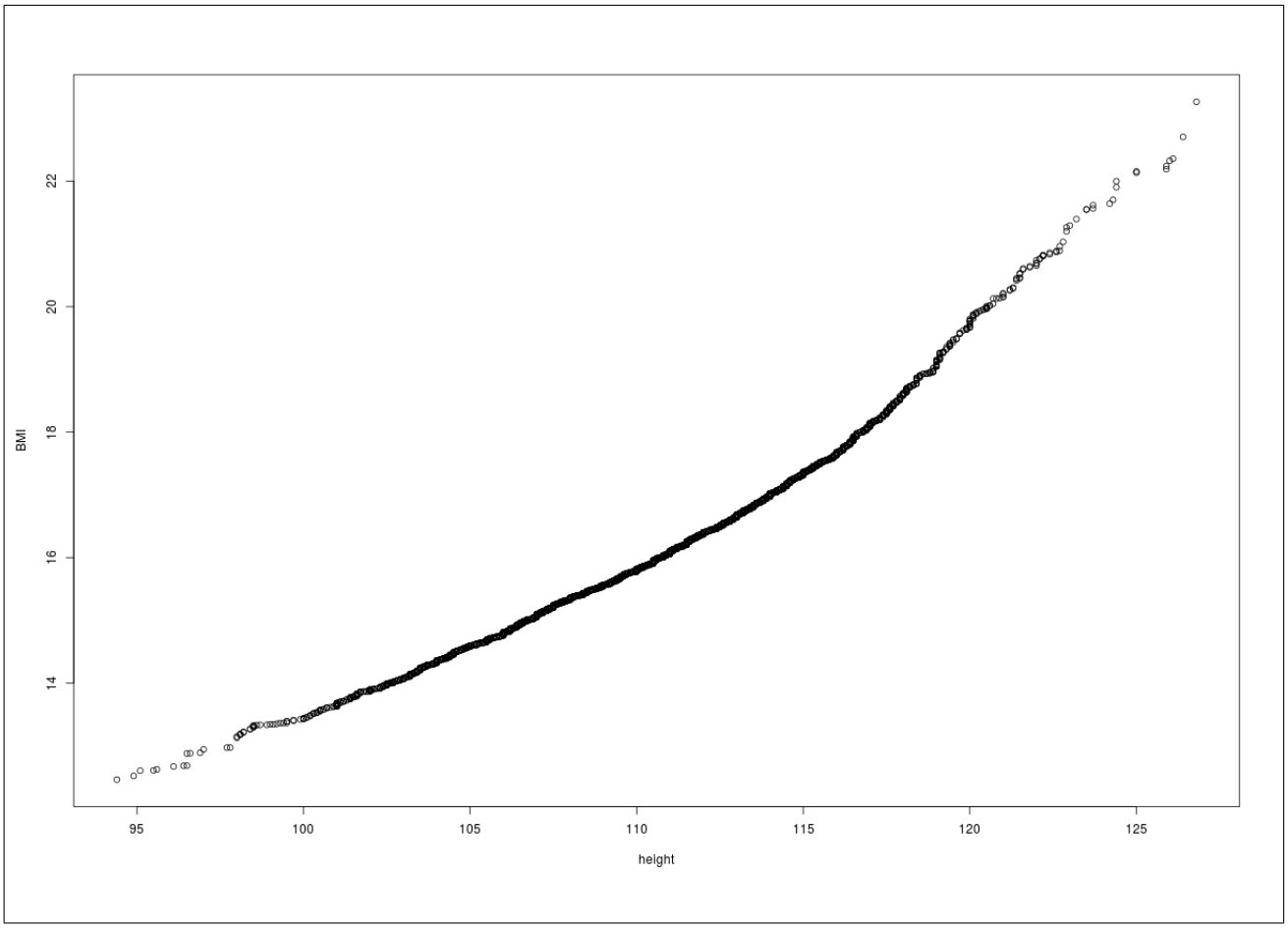


Figure 5: Male age =60.6 , height vs BMI

Comparing figure 5 with figure 6 we can notice that females has more BMI than males in the same group of height.

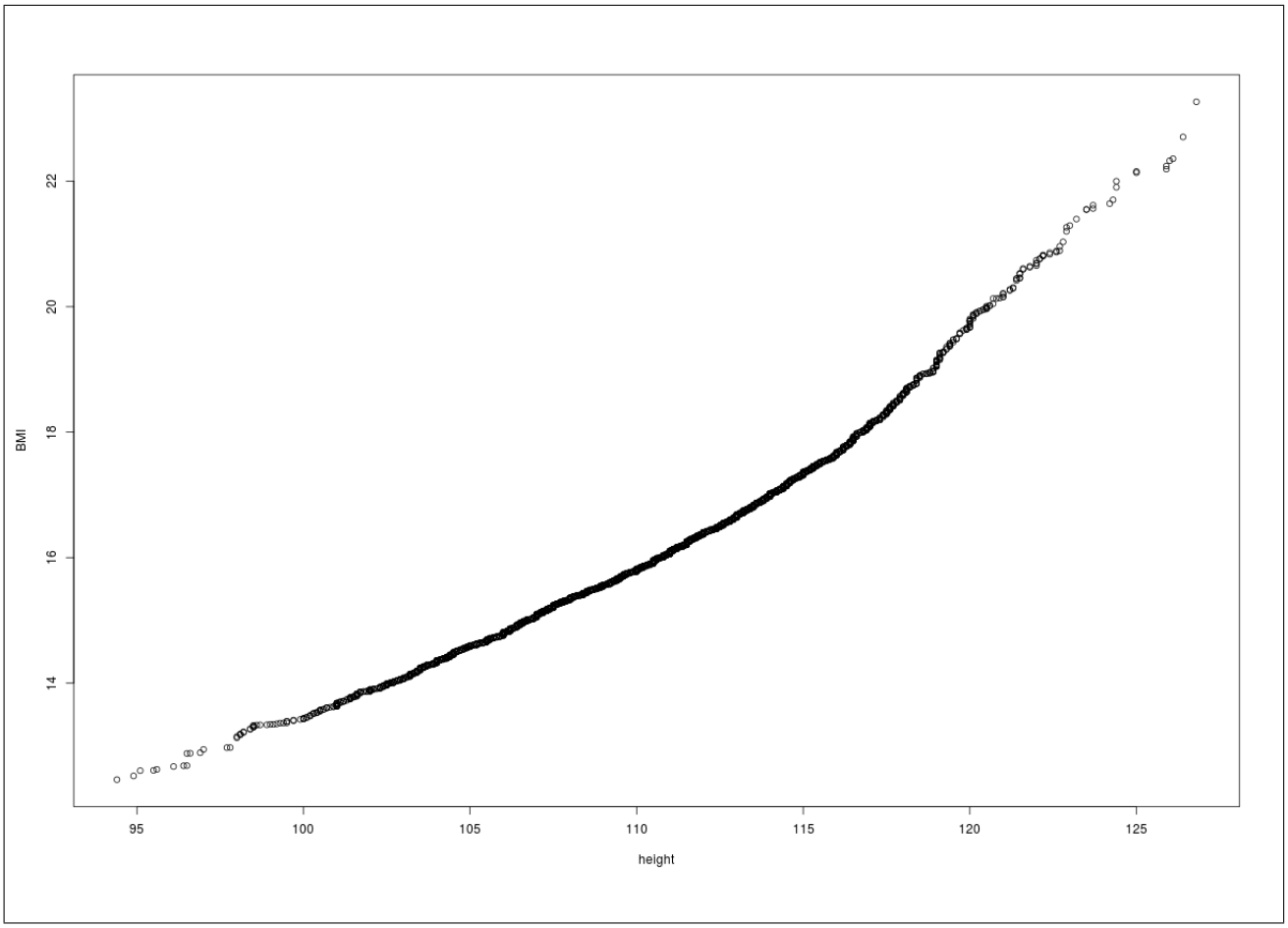


Figure 6: Female age =60.6 , height vs BMI

Third Question

For this task I took the male gender from 60.6 to 80 (about 15000 kid) for over weight children. I used the following code :

```
########Third Question ##############
overweight<-ncmp[ncmp$bmipopulationcategory=='overweight'&ncmp$genderdescription=='Male' &ncmp
    $ageinmonths>60.6 &
ncmp$ageinmonths<80,]
png('QQvsdest',width = 1200,height = 830)
qqnorm(overweight$height);qqline(overweight$height,col=2);
dev.off()

```

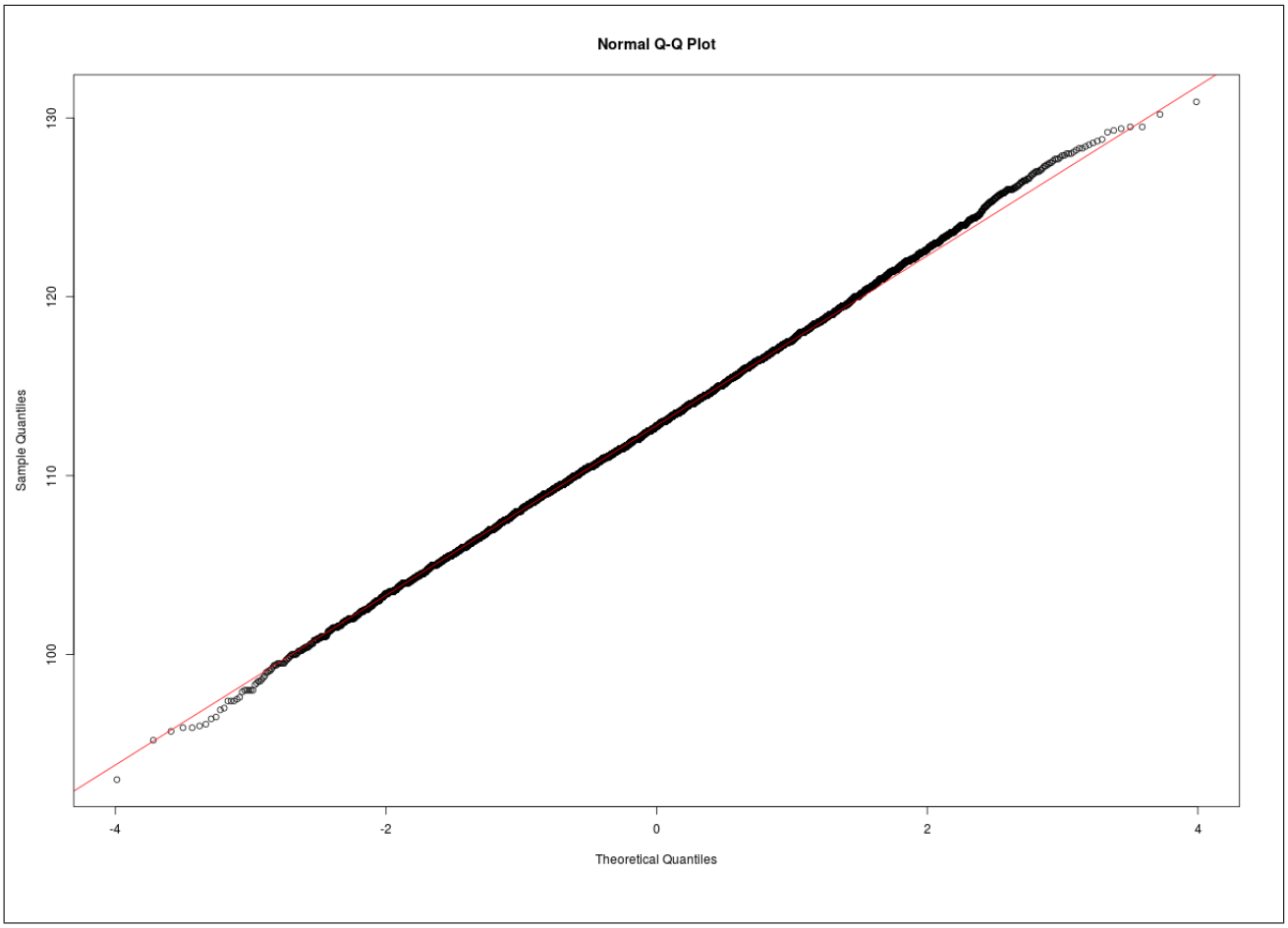


Figure 7: Overweight children height vs Theoretical normal distribution

Although figure 7 shows that the height almost identical to normal distribution but when we take all the points of overweighted children we would notice a difference.

Fourth Question

Note:I believe the question changed (it was with support more than 3 then become more than 0.3) which is the same in this question ($3/8 = 0.375$, $2/8 = 0.25$).
For this question I used python
Note: you need to generalize the last step to get all the combination we want.Here is the code :

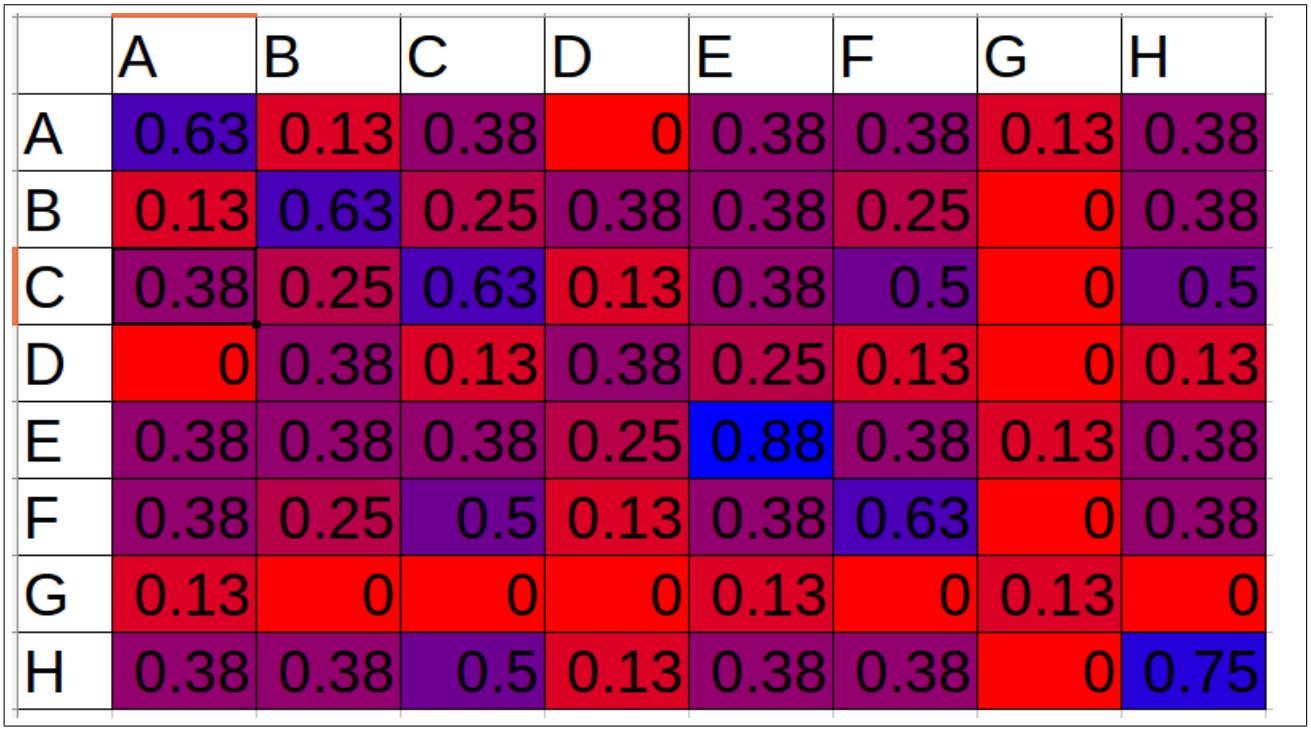
```
1
2  # coding: utf-8
3
4  # In[95]:
5
6  import itertools
7
8
9  # In[96]:
10
11 main =[]
12 main.append(sorted(['B','C', 'A' ,'F', 'H']))
13 main.append(sorted(['F', 'E', 'C', 'H']))
14 main.append(sorted(['E' ,'D', 'B']))
15 main.append(sorted(['A', 'C' ,'H', 'F']))
16 main.append(sorted(['E', 'F' ,'A']))
17 main.append(sorted(['D', 'H' ,'B']))
18 main.append(sorted(['E', 'C' ,'F', 'B', 'D']))
19 main.append(sorted(['A', 'H' ,'C', 'E']))
20 main.append(sorted(['G', 'A', 'E']))
21 main.append(sorted(['B', 'H', 'E']))
22
23
24 # In[97]:
25
```

```
lst={}
def addelement(element):
if element not in lst.keys():
lst[element]=1
else:
lst[element]+=1


# In[98]:

for i in main:
for j in i:
addelement(j)


# In[99]:

print len(lst.keys())
for key in lst.keys():
print key,lst[key]
if lst[key]<3:
lst.pop(key,None)
print '------ after cleaning--------'
for key in lst.keys():
print key,lst[key]
print len(lst.keys())


# In[148]:

lst2items={}
def addelement2(el):
el = ''.join(map(str.strip,el))
if el not in lst2items.keys():
lst2items[el]=1
else:
lst2items[el]+=1


# In[149]:

def checkcontain(myelements,thelist):
for i in myelements:
if i.strip() not in thelist:
return False
return True


# In[150]:

newcomb = itertools.combinations(lst.keys(), 3)

for i in newcomb:
for j in main:
if checkcontain(i,j):
addelement2(i)


# In[151]:

for key in lst2items.keys():
print key,lst2items[key]
if lst2items[key]<3:
lst2items.pop(key,None)
print '------ after cleaning--------'
for key in lst2items.keys():
print key,lst2items[key]
```

You can find a copy on git hub here DM in Home work 05 folder.
I started with one element : A 5 C 5 B 5 E 7 D 3 F 5 H 6
Then for 2 elements : BD 3 BE 3 AC 3 AE 3 EH 3 AF 3 AH 3 EF 3 BH 3 CF 4 CE 3 CH 4 FH 3
Then for 3 elements: ACH 3 CFH 3
And there is no 4 elements with support 3 or more.

Fifth Question

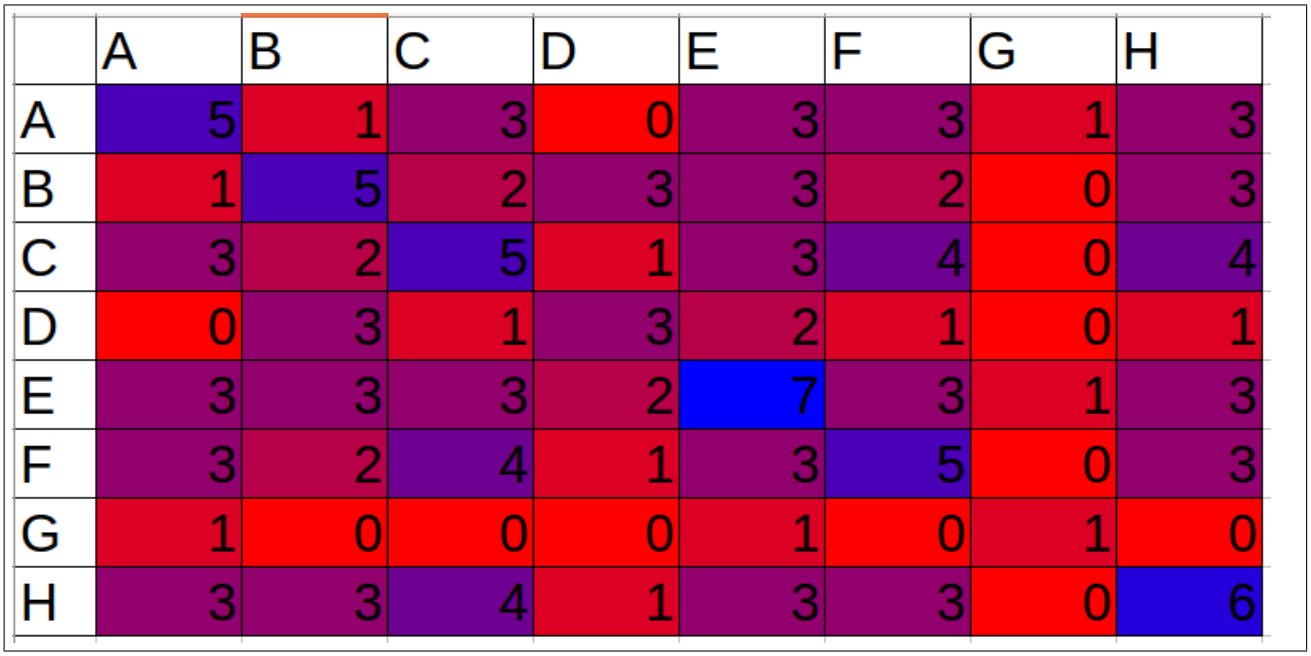
For this question I put all the information in Calc sheet and fulled it with the results I got from 4th question. Here is the heat map:



	A	B	C	D	E	F	G	H
A	0.63	0.13	0.38	0	0.38	0.38	0.13	0.38
B	0.13	0.63	0.25	0.38	0.38	0.25	0	0.38
C	0.38	0.25	0.63	0.13	0.38	0.5	0	0.5
D	0	0.38	0.13	0.38	0.25	0.13	0	0.13
E	0.38	0.38	0.38	0.25	0.88	0.38	0.13	0.38
F	0.38	0.25	0.5	0.13	0.38	0.63	0	0.38
G	0.13	0	0	0	0.13	0	0.13	0
H	0.38	0.38	0.5	0.13	0.38	0.38	0	0.75

Figure 8: Heatmap for support

To calculate the confidence I actually used a trick :) .In figure 9 you can see the table that I depended on to calculate both (support & confidence).



	A	B	C	D	E	F	G	H
A	5	1	3	0	3	3	1	3
B	1	5	2	3	3	2	0	3
C	3	2	5	1	3	4	0	4
D	0	3	1	3	2	1	0	1
E	3	3	3	2	7	3	1	3
F	3	2	4	1	3	5	0	3
G	1	0	0	0	1	0	1	0
H	3	3	4	1	3	3	0	6

Figure 9: Table shows how many two items occurred

We can notice that if we divide each cell on 8 (number of transaction) we get the support and if we divide each row on the diagonal element we get the confidence.
Here is the confidence table :

	A	B	C	D	E	F	G	H
A	1.00	0.20	0.60	0.00	0.60	0.60	0.20	0.60
B	0.20	1.00	0.40	0.60	0.60	0.40	0.00	0.60
C	0.60	0.40	1.00	0.20	0.60	0.80	0.00	0.80
D	0.00	1.00	0.33	1.00	0.67	0.33	0.00	0.33
E	0.43	0.43	0.43	0.29	1.00	0.43	0.14	0.43
F	0.60	0.40	0.80	0.20	0.60	1.00	0.00	0.60
G	1.00	0.00	0.00	0.00	1.00	0.00	1.00	0.00
H	0.50	0.50	0.67	0.17	0.50	0.50	0.00	1.00

Figure 10: Confidence table

Note: All code,pictures , python , R , ipython in this repository aqeel13932