HW3

Biying Wan

1

- install.packages("digest")
- library("digest")
- digest("I learn a lot from this class when I am proper listening to the professor","sha256")
- [1]"c16700de5a5c1961e279135f2be7dcf9c187cb6b21ac8032308c715e1ce9964c"
- digest("I do not learn a lot from this class when I am absent and playing on my Iphone","sha256")
- [1]"2533d529768409d1c09d50451d9125fdbaa6e5fd4efdeb45c04e3c68bcb3a63e"

2 DSA (Digital Signature Algorithm)

- A **digital signature** is basically a way to ensure that an electronic document (e-mail, spreadsheet, text file, etc.) is **authentic**. Digital signatures are used to verify that a message or document was authored by a certain person, and that it was not altered or modified by anyone else.

- One of the most common digital signature mechanisms is DSA. The **Digital Signature Algorithm (DSA)** is the basis of the **Digital Signature Standard (DSS)**, a U.S. Government document.

2 DSA (Digital Signature Algorithm)

- DSA lets one person with a secret key “sign” a document, so that others with a matching public key can verify it must have been signed only by the holder of the secret key.

- Digital signatures depend on **hash functions**, which are one-way computations done on a message. They are called “one-way” because there is no known way (without infeasible amounts of computation) to find a message with a given hash value. The result has a fixed length, which is 160 bits in the case of the Secure Hash Algorithm (SHA) used by DSA.

2 DSA (Digital Signature Algorithm)

- In practice, digital signatures are used to sign the hash values of messages, not the messages themselves. Thus it is possible to sign a message's hash value, without even knowing the content of the message. This makes it possible to have *digital notaries*, who can verify a document existed (and was signed), without the notary knowing anything about what was in the document.

3

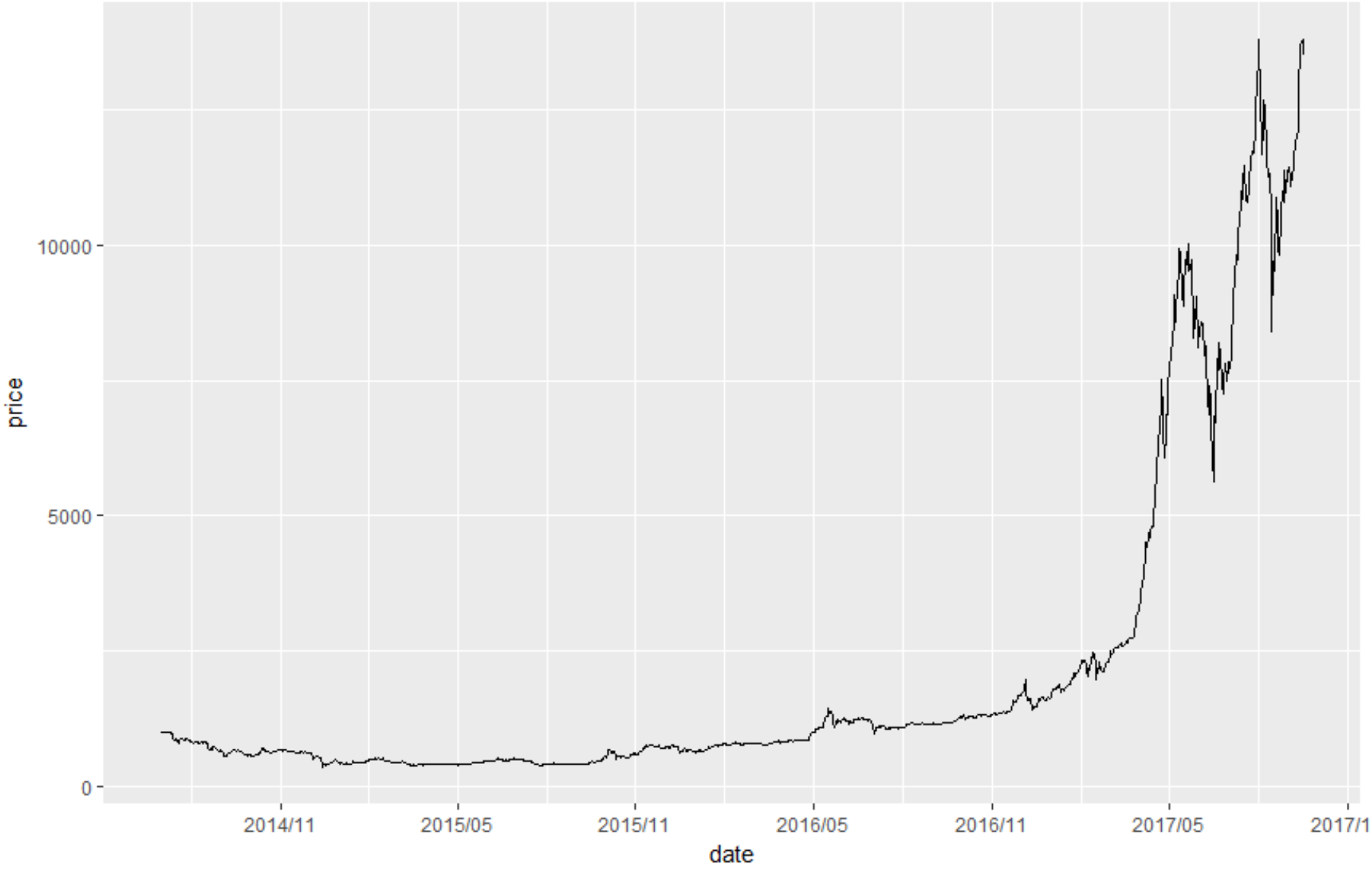
- install.packages("rjson")
- library("rjson")
- json_file3="D:/研二/研二上/大数据与互联网金融/HW3/test.json"
- json_data3<- fromJSON(paste(readLines(json_file3), collapse=""))
- json_data3<- as.data.frame(json_data3)
- print(json_data3)

```
name email gender hobby name.1 email.1 gender.1 hobby.1
lucy   @01   male  surf    lim     @02      male    surf
lucy   @01   male  ball    lim     @02      male    ball
```

4.1

- install.packages("ggplot2")
- install.packages("scales")
- library("rjson")
- json_file="D:/研二/研二上/大数据与互联网金融/HW3/crix.json"
- json_data <- fromJSON(paste(readLines(json_file), collapse=""))
- json_df <- as.data.frame(c(json_data[[1]][1],json_data[[1]][2]))
- for (i in 2:length(json_data)){
- json_df <- rbind(json_df,as.data.frame(c(json_data[[i]][1],json_data[[i]][2])))
- }
- json_df$date <- as.POSIXct(json_df$date)
- library(ggplot2)
- library(scales)
- ggplot(json_df)+
- geom_line(aes(x=date,y=price))+
- scale_x_datetime(breaks=date_breaks("6 month"),labels=date_format("%Y/%m"))

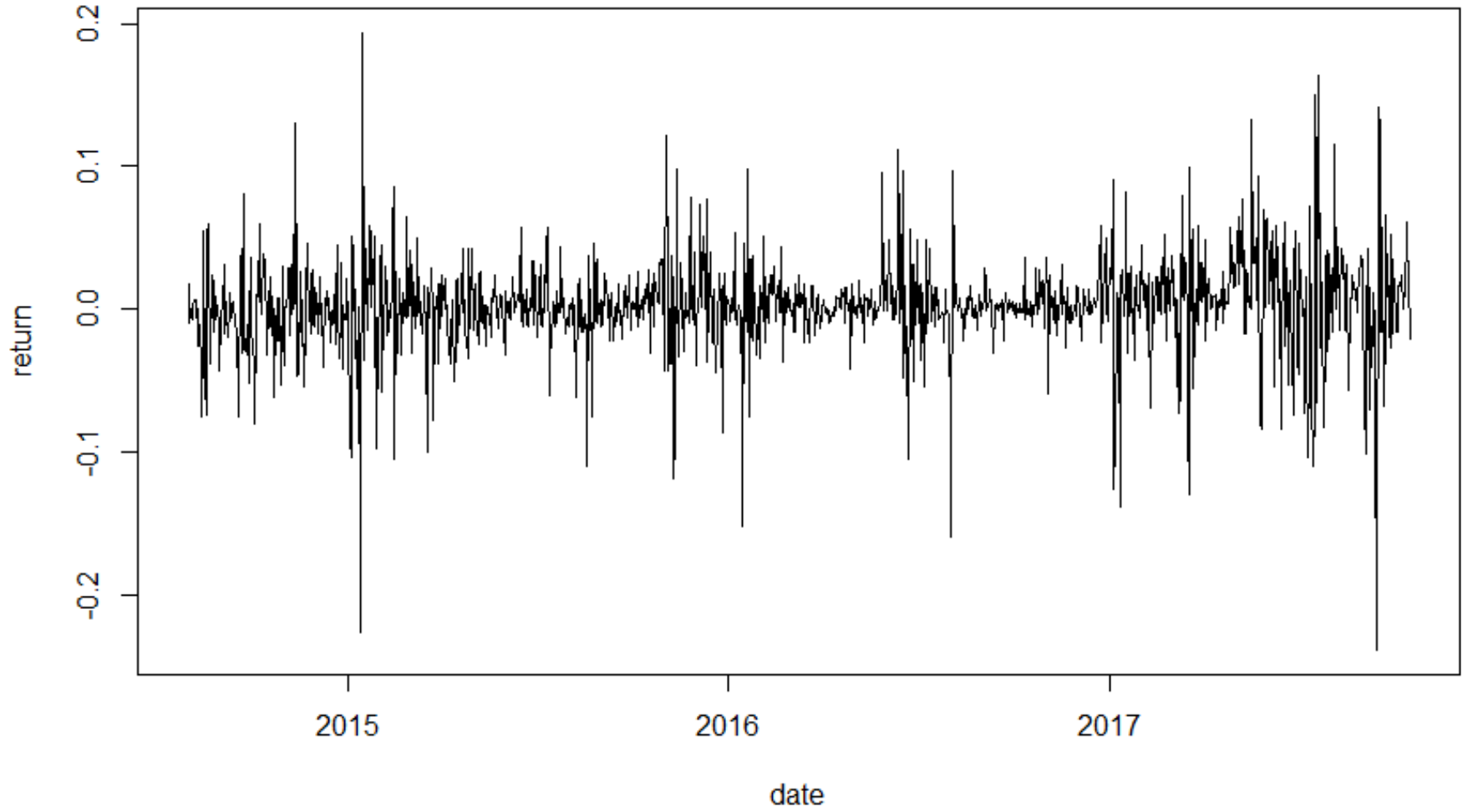
4.1 Price of CRIX



4.2

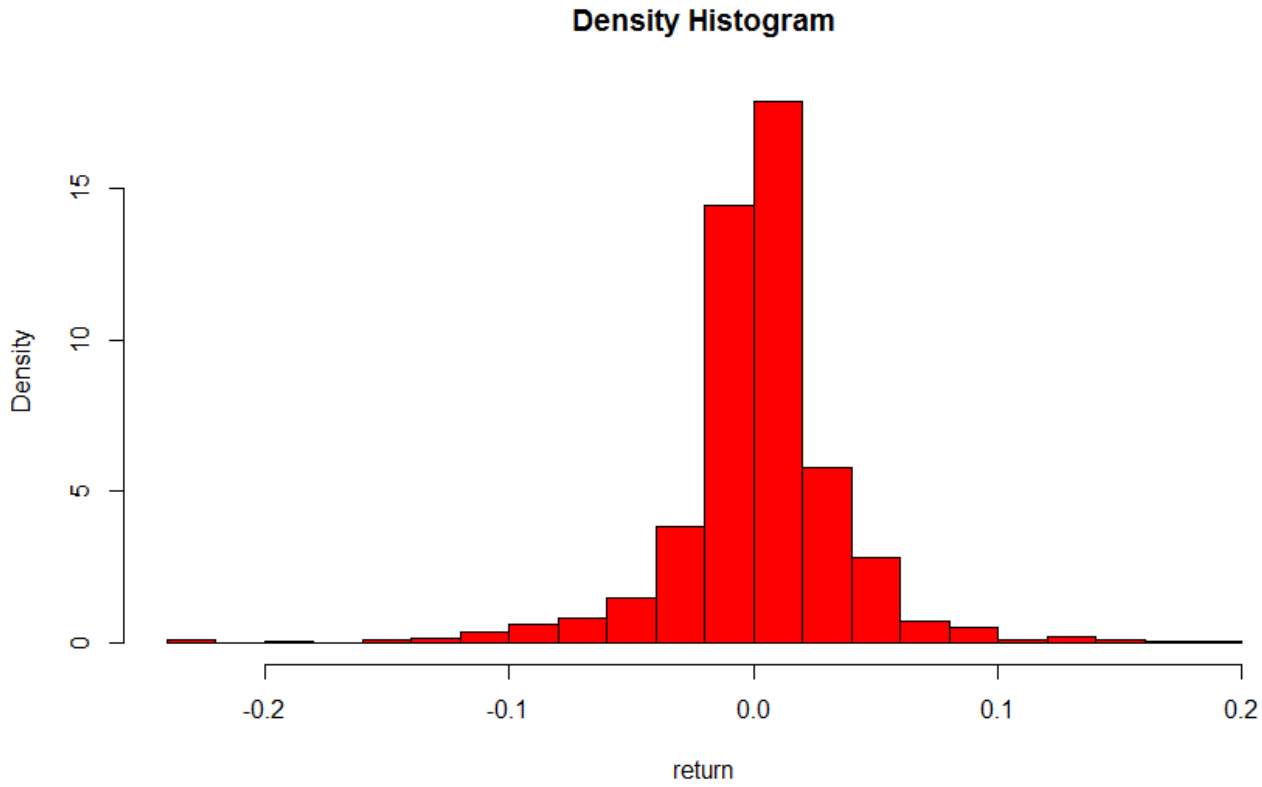
- x<-json_df[,2]
- return<-log(x[2:nrow(json_df)])-log(x[1:nrow(json_df)-1])
- return<-c(NA,return)
- json_return<-as.data.frame(cbind(json_df,return))
- json_return<-json_return[,-2]
- plot(json_return,type="l")

4.2 Stochastic Property of CRIX



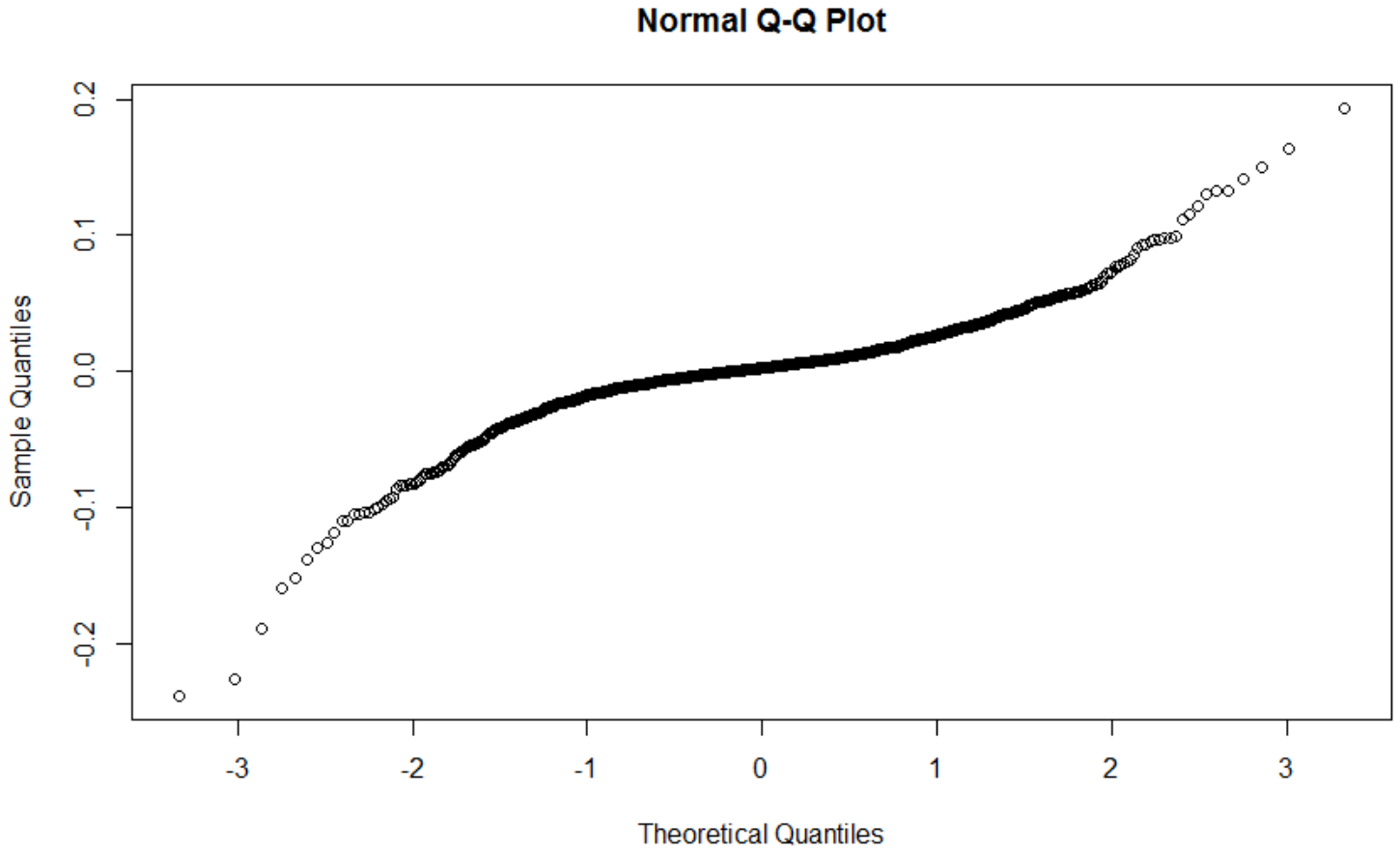
4.3 Distributional Properties of CRIX

- hist(json_return$return,freq=FALSE,breaks=12,col="red",xlab="return",main="Density Histogram")



4.3 Distributional Properties of CRIX

- qqnorm(json_return$return)



CRIX returns is not normally distributed.