final_project

Han Xiao

4/30/2018

data prep

This is the data preprocessing code.

```
library(tidyverse)
```

```
## -- Attaching packages ----------------------------------------- tidyverse 1.2.1 --
```

```
## ✓ ggplot2 2.2.1     ✓ purrr   0.2.4
## ✓ tibble  1.4.2     ✓ dplyr   0.7.4
## ✓ tidyr   0.8.0     ✓ stringr 1.3.0
## ✓ readr   1.1.1     ✓ forcats 0.3.0
```

```
## -- Conflicts -------------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(kernlab)
```

```
##
## Attaching package: 'kernlab'
```

```
## The following object is masked from 'package:purrr':
##
##     cross
```

```
## The following object is masked from 'package:ggplot2':
##
##     alpha
```

```
library(anytime)
library(rlist)
#get data
data <- read.csv("bitstampUSD_1-min_data_2012-01-01_to_2018-03-27.csv")
alldata <- data.frame(data)
#this is tbe original data
tail(alldata)
```

```
##          Timestamp    Open    High     Low   Close Volume_.BTC.
## 3273372 1522108500 8167.70 8168.00 8161.11 8161.11    0.6313703
## 3273373 1522108560 8159.21 8159.39 8153.62 8153.68    0.6631919
## 3273374 1522108620 8159.38 8163.72 8152.81 8155.00    3.7200237
## 3273375 1522108680 8157.47 8158.60 8130.01 8130.01   27.2944358
## 3273376 1522108740 8145.17 8149.66 8130.01 8149.66   23.0652813
## 3273377 1522108800 8152.26 8164.68 8151.37 8153.04   17.7581301
##         Volume_.Currency. Weighted_Price
## 3273372          5156.590       8167.299
## 3273373          5408.215       8154.827
## 3273374         30333.753       8154.183
```

```
## 3273375        222388.762       8147.769
## 3273376        187570.785       8132.170
## 3273377        144820.147       8155.146
```

```
alldata$date <- anydate(data$Timestamp)

dailyprice <- aggregate(alldata$Weighted_Price, list(alldata$date), mean)
write.csv(x = dailyprice,file = "shorter.csv",row.names=FALSE)
dailyprice <- read.csv("shorter.csv")
#this is the processed data
tail(dailyprice)
```

```
##         Group.1        x
## 2270 2018-03-21 8993.399
## 2271 2018-03-22 8714.325
## 2272 2018-03-23 8587.795
## 2273 2018-03-24 8800.013
## 2274 2018-03-25 8516.663
## 2275 2018-03-26 8164.808
```

```
#split test and train
train_begin <- c("2016-01-01")
train_end <- c("2018-01-25")
test_begin <- c("2018-02-26")
test_end <- c("2018-03-26")

train_begin_date <- as.Date(train_begin)
train_end_date <- as.Date(train_end)
test_begin_date <- as.Date(test_begin)
test_end_date <- as.Date(test_end)

alldays <- as.numeric(test_end_date-train_begin_date)

#function help process lookback data
create_lookback <- function(dataframe, lookback){
  list1 <- list()
  list2 <- list()
  for (i in c(1:(length(dataframe)-lookback))){
    a <- list(dataframe[i:((i+lookback)-1)])
    list1[length(list1)+1] <- a
    #list.append(list1,X=c(a))
    #print(list1)
    b <- dataframe[i+lookback]
    list2[length(list2)+1] <- b

  }

  df <- as.data.frame(t(as.data.frame(list1)))
  #print(df)
  df$Y <- unlist(list2)
  return(df)
}


traindays <- as.numeric(train_end_date-train_begin_date)
```

```
traindays
```

```
## [1] 755
```

```
testdays <- as.numeric(test_end_date-test_begin_date)
testdays
```

```
## [1] 28
```

```
lookback <- 2
traindata <- dailyprice[(nrow(dailyprice)-alldays):(nrow(dailyprice)-alldays+traindays),]
df_train <- create_lookback(traindata$x,lookback)
#names(df_train)[1] <- c("Y")
testdata <- dailyprice[(nrow(dailyprice)-testdays):(nrow(dailyprice)),]
df_test <- create_lookback(testdata$x,lookback)
testdata <- testdata[(lookback+1):nrow(testdata),]
#names(df_test)[1] <- c("Y")
train_test_data <- dailyprice[(nrow(dailyprice)-alldays):(nrow(dailyprice)),]
#df_all <- create_lookback(train_test_data$x,lookback)
```

The data is downloaded from a kaggle challenge It contains price data and date from 2011 to 2018 of bitcoin. I already preprocess the data and split those into test and train data. I only selected 2016-01-01 to2018-01-25 as trianing data. 2018-01-26 to 2018-03-26 as test data. I created a lookback function.Lookback is the number of days of the prices you want it to be the factors to predict the current day's price. if the look back is set to 2, it means that one day's price will have previous two days' price as X for training to predict the price of that day. A picture of lookback=2 is shown in figure 1

For the graphs, black points are the training data, blue points are the testing data true values, red line connects the predicted prices. The mean square error of each model can show the how good the model is. There are definitely more and better models to do this. Also there can be more factors to be concerned to predict the price. As far as I know, LSTM is a better choice in deep learning to do a forcasting in time series.

```
#lm_fit <- lm(Weighted_Price~.,traindata)
lm_fit <- lm(Y~.,df_train)
lm_fit
```

```
##
## Call:
## lm(formula = Y ~ ., data = df_train)
##
## Coefficients:
## (Intercept)           V1           V2
##     12.2304      -0.2076       1.2073
```

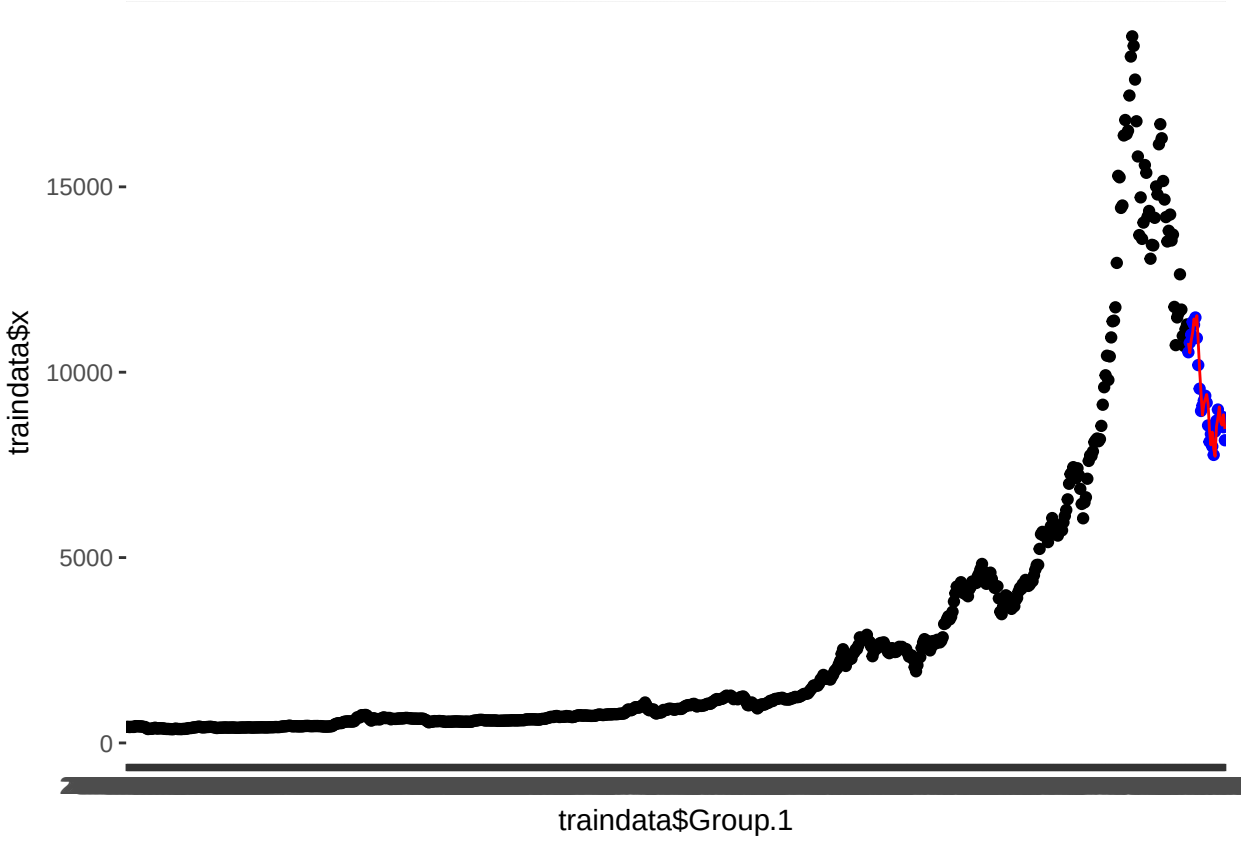
```
lm_predictions <- predict(lm_fit, df_test)
mse <- mean((testdata$x - lm_predictions)^2)
print(mse)
```

V1	V2	Y
433.1269	432.7755	429.1828
432.7755	429.1828	432.9407
429.1828	432.9407	431.7050
432.9407	431.7050	431.5506
431.7050	431.5506	453.1439
431.5506	453.1439	453.1145
453.1439	453.1145	448.8666
453.1145	448.8666	445.9814
448.8666	445.9814	446.0317
445.9814	446.0317	442.9236
446.0317	442.9236	430.2935
442.9236	430.2935	428.7694

Figure 1:

```
## [1] 121890.2
```

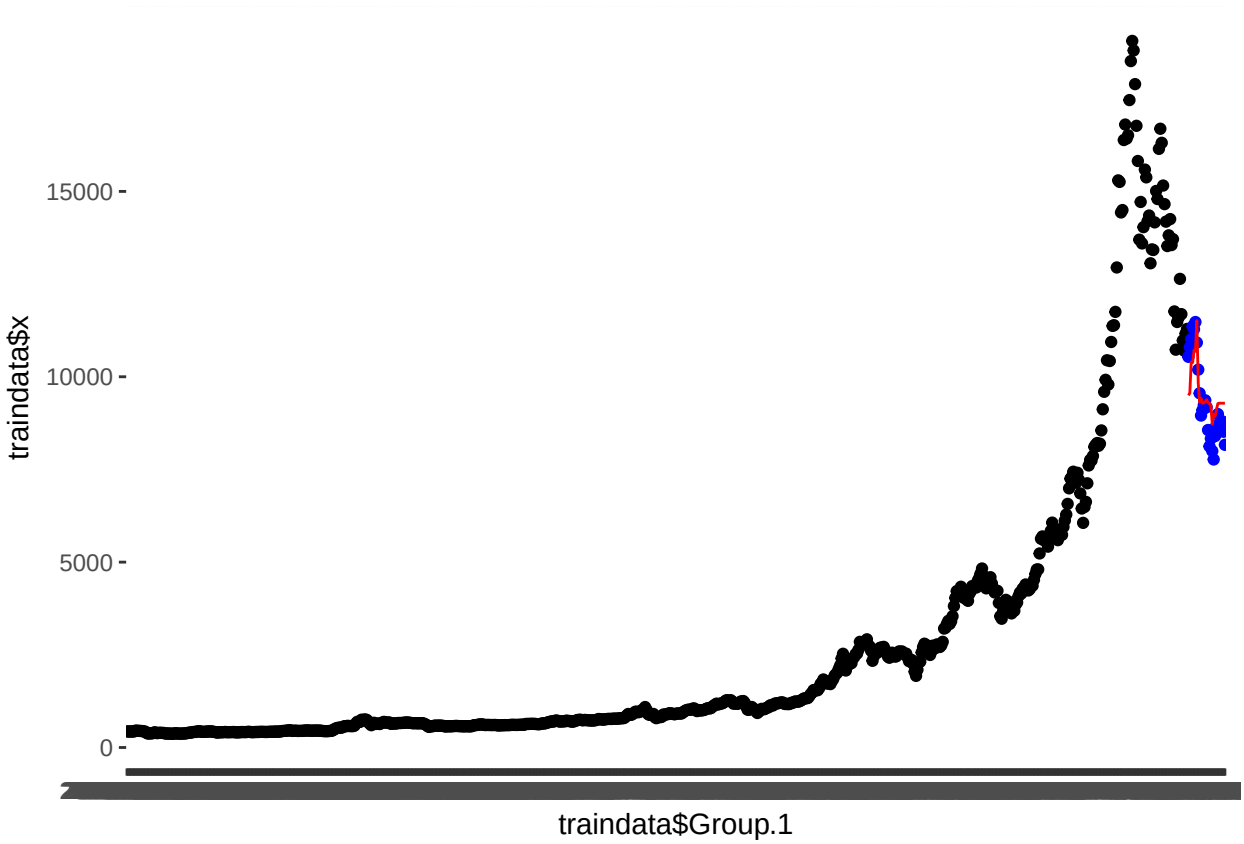
```
prediction_data <- tail(testdata,length(lm_predictions))
ggplot()+geom_point(data = traindata,aes(x=traindata$Group.1,y=traindata$x),color="black")+geom_point(da
```



SVMModel

```
## Support Vector Machine object of class "ksvm"
##
## SV type: eps-svr  (regression)
##  parameter : epsilon = 0.1  cost C = 1
##
## Gaussian Radial Basis kernel function.
##  Hyperparameter : sigma =  210.298929060561
##
## Number of Support Vectors : 145
##
## Objective Function Value : -65.0705
## Training error : 0.022628
```

```
## [1] 512119
```


15000
10000
5000
0
traindata$x
traindata$Group.1