**Write-Up**

**Dataset I used and Why**

**I downloaded the data from Basketball Reference of which stating four explicit variables that could help me better understanding the team’s performance. I’m a die heart fan of Golden State Warriors way before they turned into 3 pointers army. I’ve heard a lot of critiques from other NBA enthusiasts judging the way that 3 pointers have strong influence on winning Game in the finals and Warriors is on top of that. I really want to overthrow these prejudice by comparing solely using 3 pointers and four important elements in a basketball game (total Rebounds, Assist, three pointers and Blocks).**

**Classification Method used and Reason for choosing it**

**For classification methodology, I chose QDA. The reason for utilizing it over LDA is because it has a more flexibility than LDA for prediction or forecasting. And it’s relatively simply to use than that of SVM, I understand the concept of SVM but having trouble on coding it in R when I’m trying to use the non-linear classification in SVM. Fortunately, I found my dataset (Finals statistics for Golden State Warrriors) fits the linear model in QDA.**

**Difficulty Encountered and Solution**

**Problem**

**According to the ISL text for QDA and online resource, there’s barely a chance for me to use cross validation in QDA**

**Solution: I shuffle the data and put them Train and Test instead.**

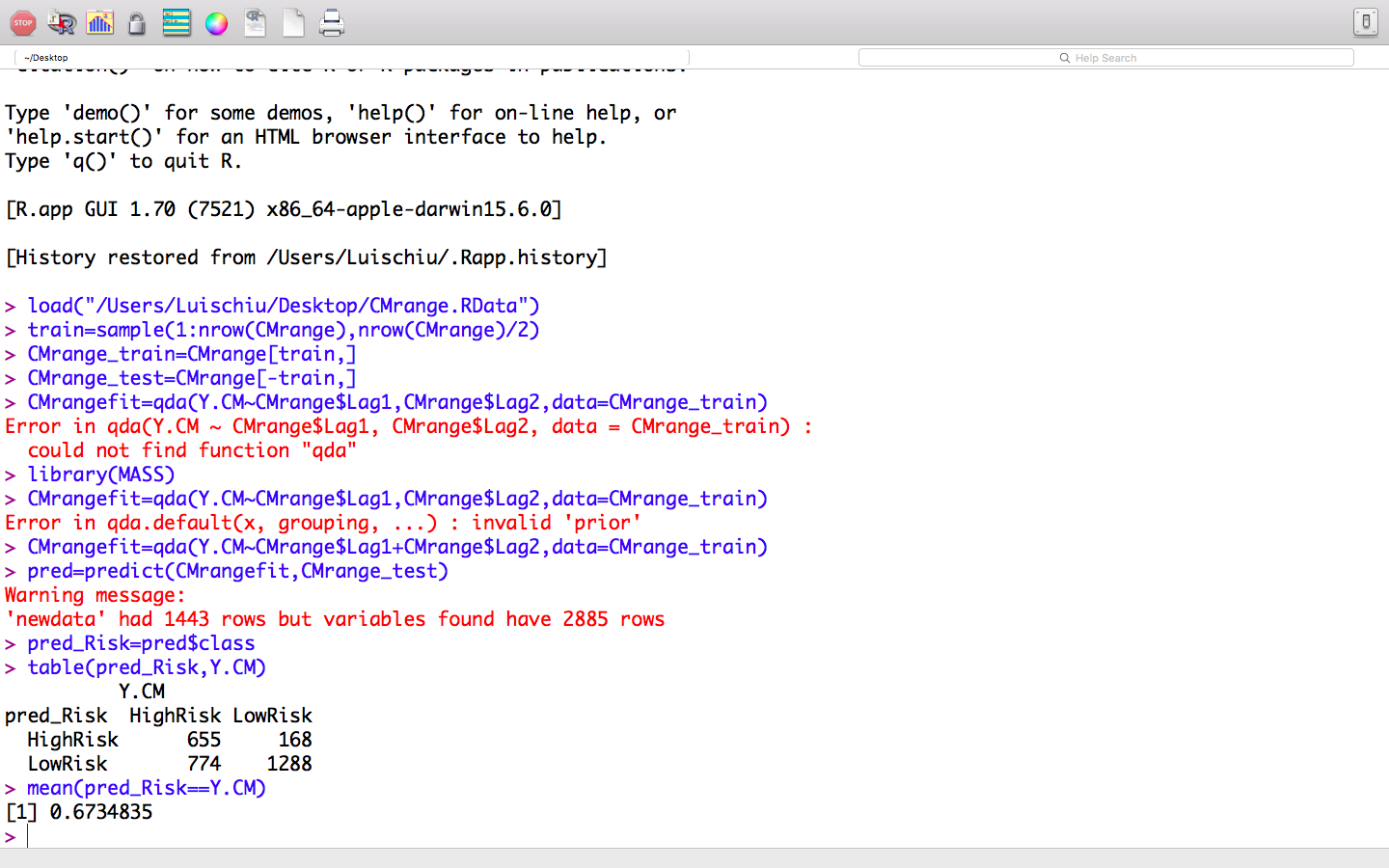
**Opting-out outliers**

**Solution: In a basketball game, I think the data set I used is relatively small and close to each other or I can say there’s no particular outliers in the dataset while I was entering them one by one into Excel in csv form.**

**Warning message while using QDA function**

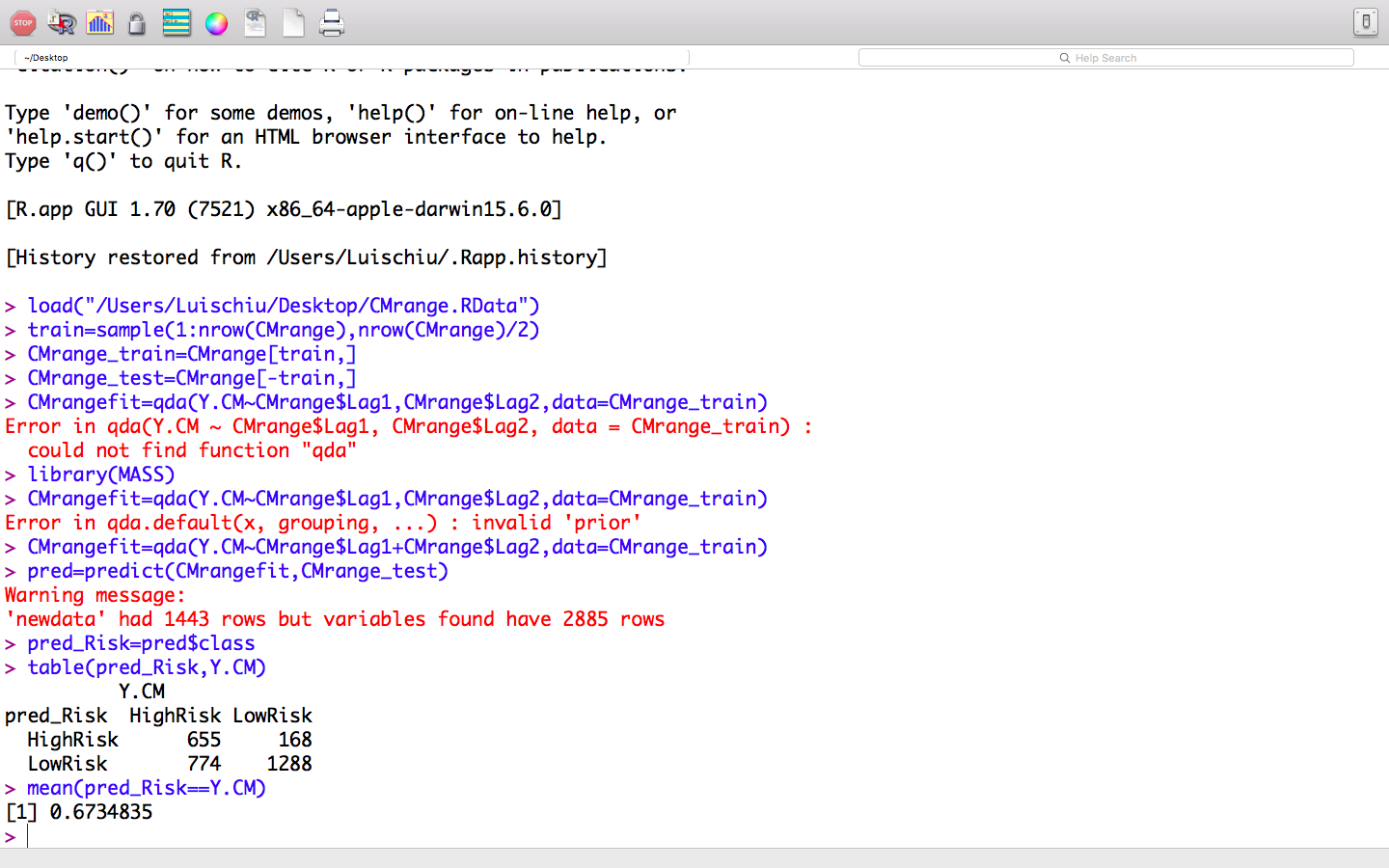
**QDA for Stock Data**

**Step 1: Simple Random sampling**



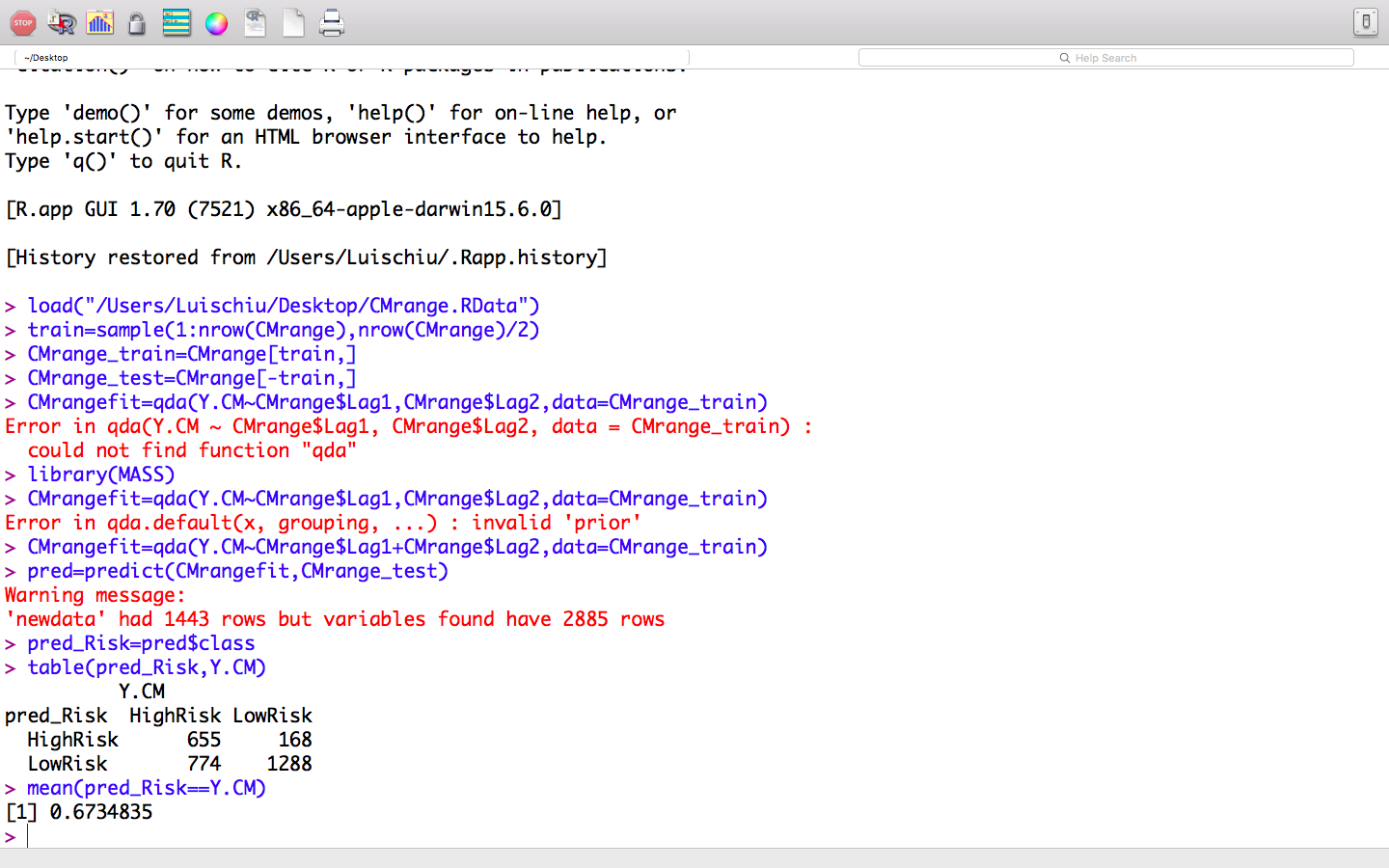
Answer: I used simple random sampling by applying the sample function since I could not find a way to do cross validation which, otherwise, would make the dataset more fit to the model. And here, I used 50% of my data which consisted of 1442 as my Train dataset and remaining as my Test Dataset.

**Step 2 : Using QDA as model for forecasting**

****

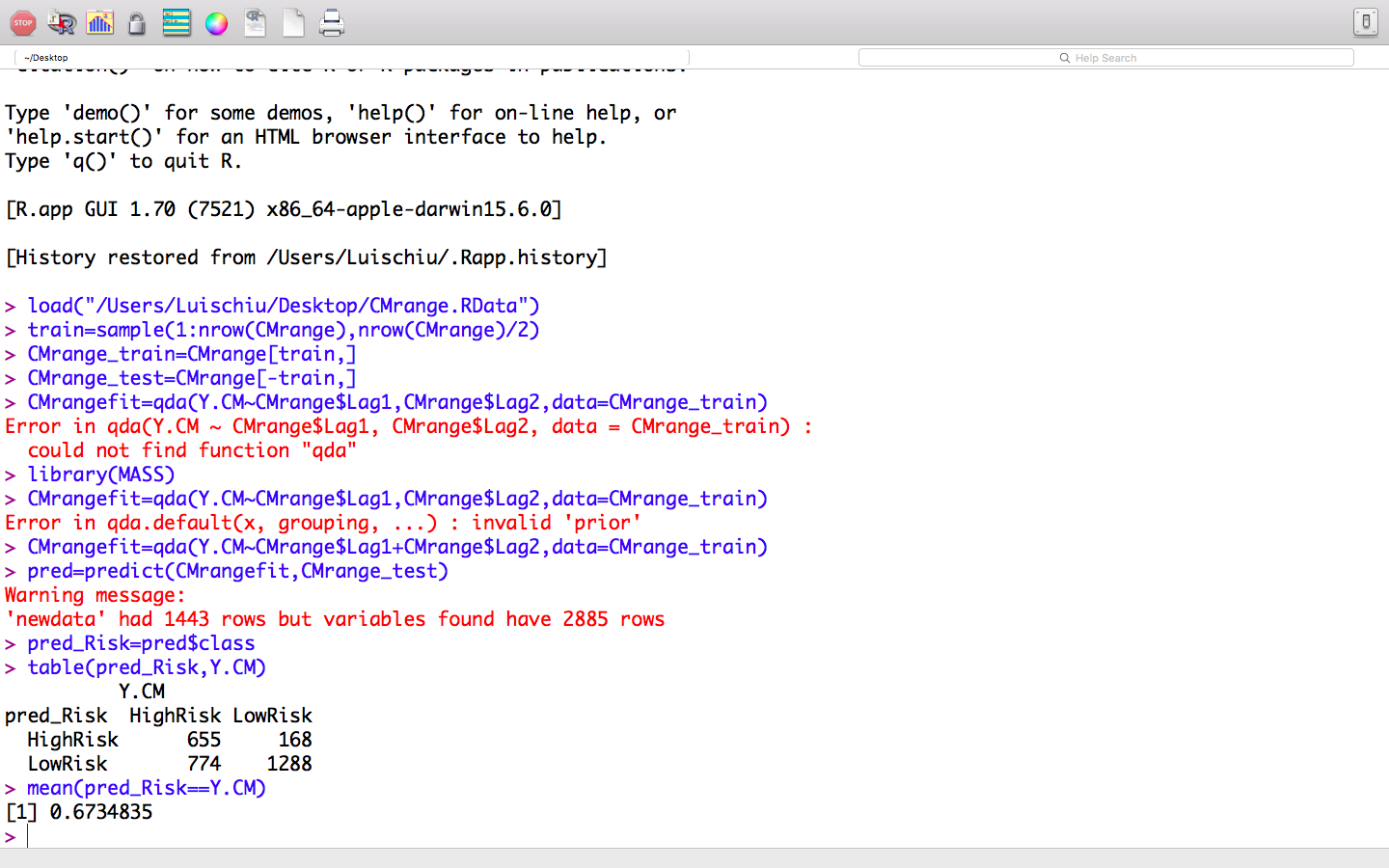
Answer: I used library (MASS) to called out QDA function in R and then I used Y.CM which is the class I used which catergorized as “HighRisk” and “LowRisk” (Something We did in our 3920 class) and use 2 variables Lag1 and Lag2 as my x-variables to complete the model for prediction of test data.

**Step 3: Applying the QDA classification to Test data**

****

Answer: After generating the QDA classification method with my Train data into variable call “CMrangefit”, then I use predict() function to applying the classification method to my test data to see if it’s working and the percentage of successful prediction.

**Step 4: Compare Prediction with the actual data**

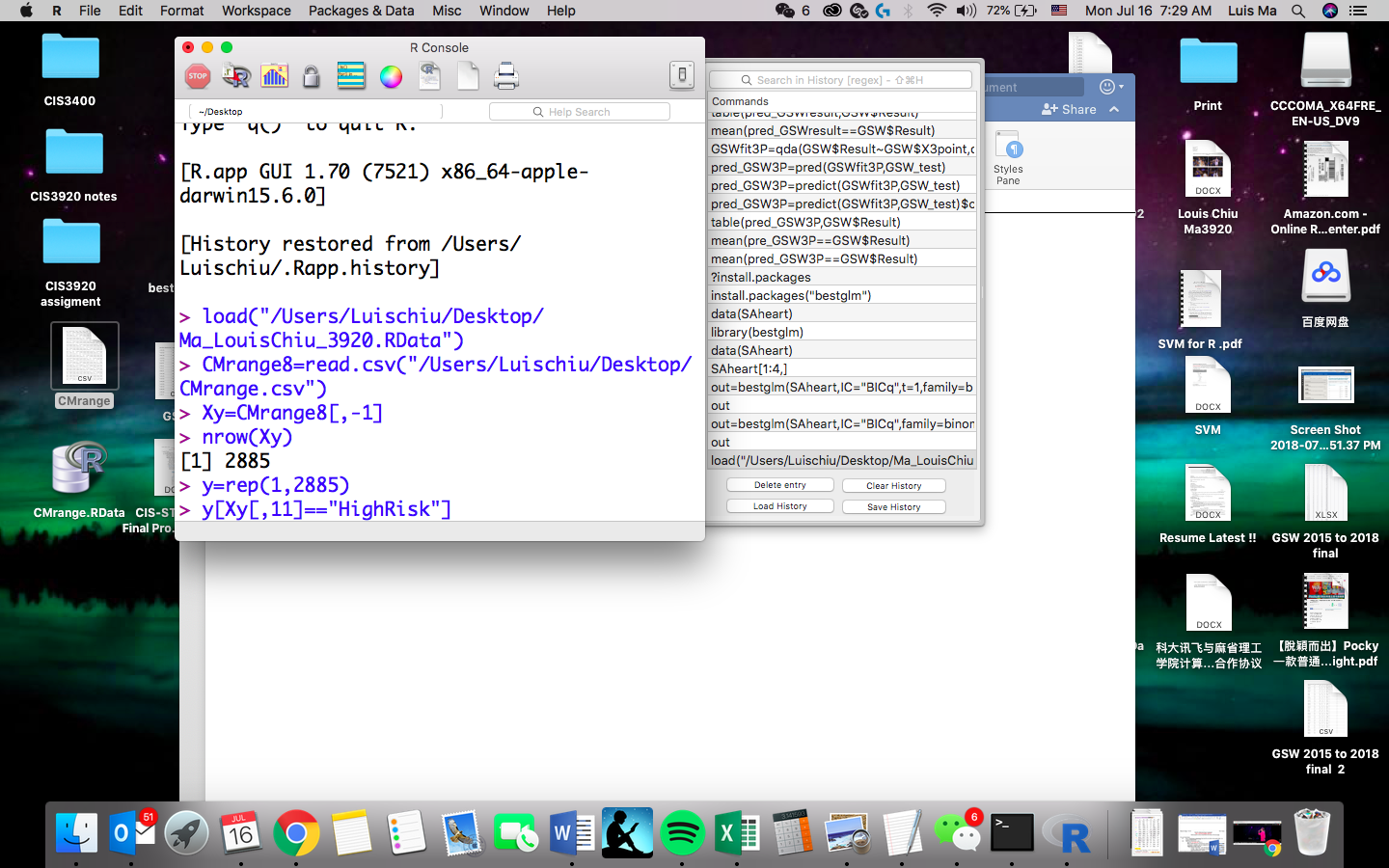
**

Answer: I used table function to do the comparison between “Y.CM”(Actual Class) in my dataset and “Pred\_Risk”(Prediction by using QDA), the diagnosis showed the successful prediction. The I calculate the percentage of successful prediction by finding the average of number of successful prediction from Pred\_Risk and that of the actual class (Y.CM) . It’s giving a fair prediction rate about 67%.

*Reference: ISL text, P.163*

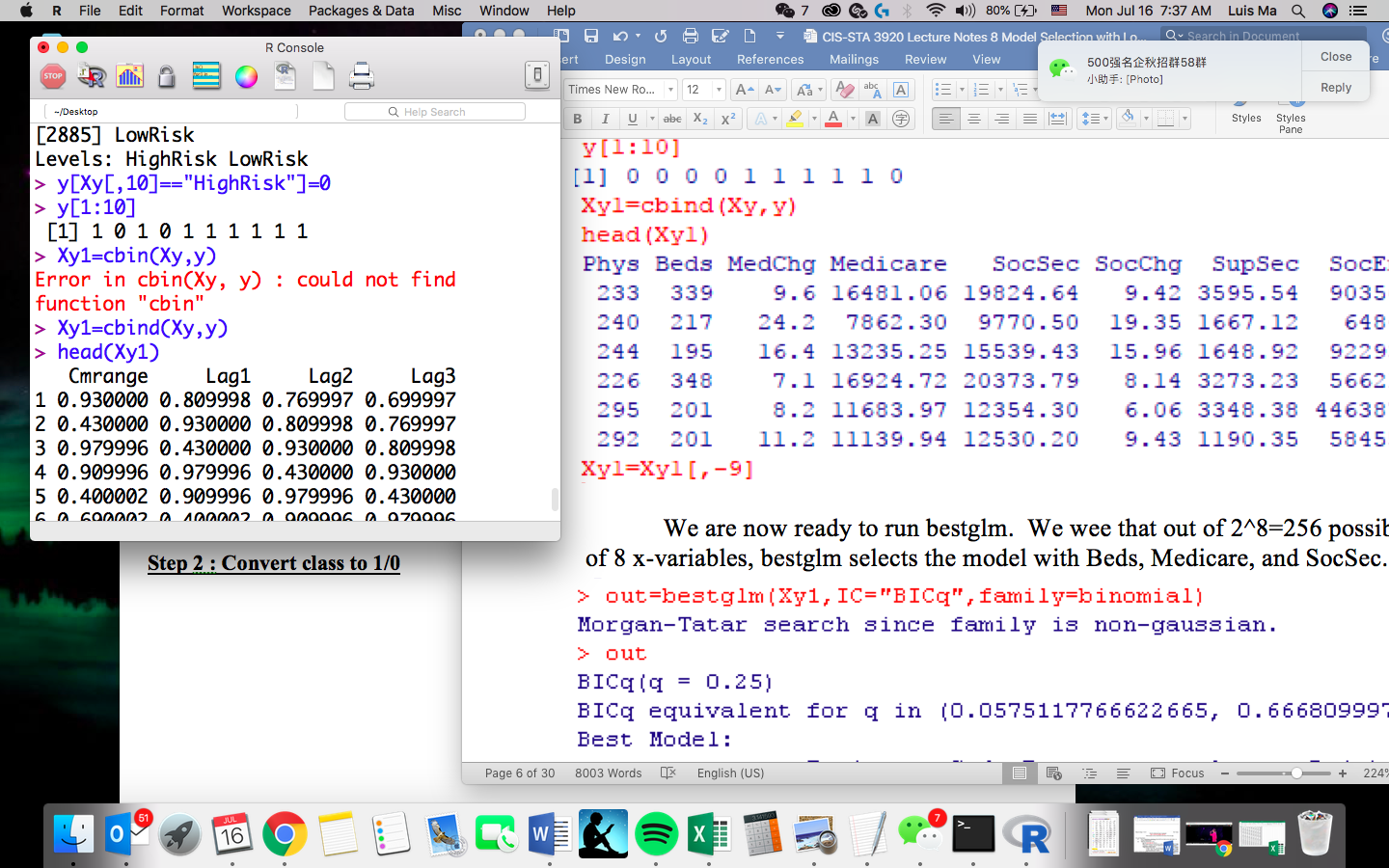
**Bestglm for Stock Range Data**

**Step 1: Create Vector**

******

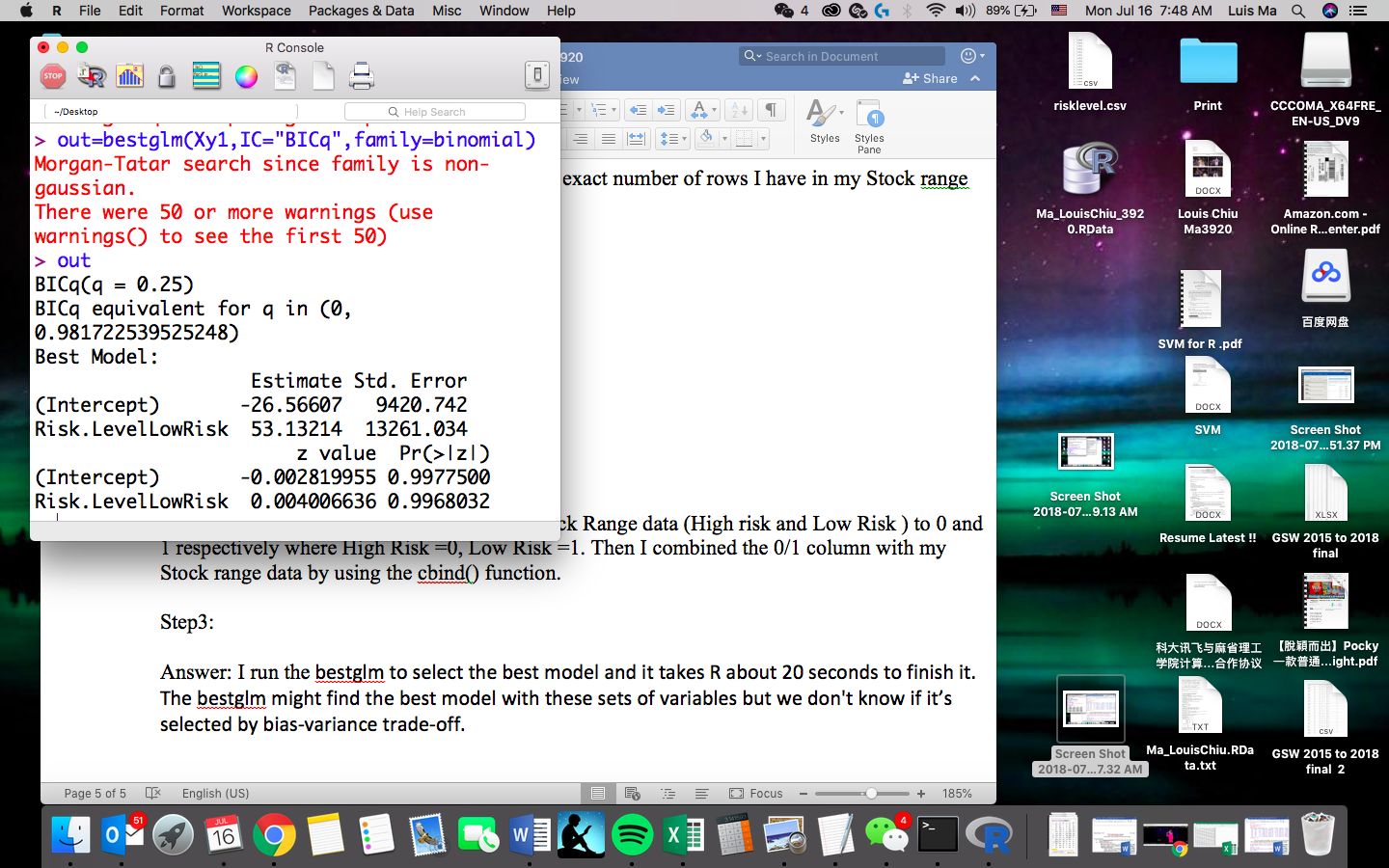
Answer: I created a vector with total 2885, the exact number of rows I have in my Stock range Data and fill with 1.

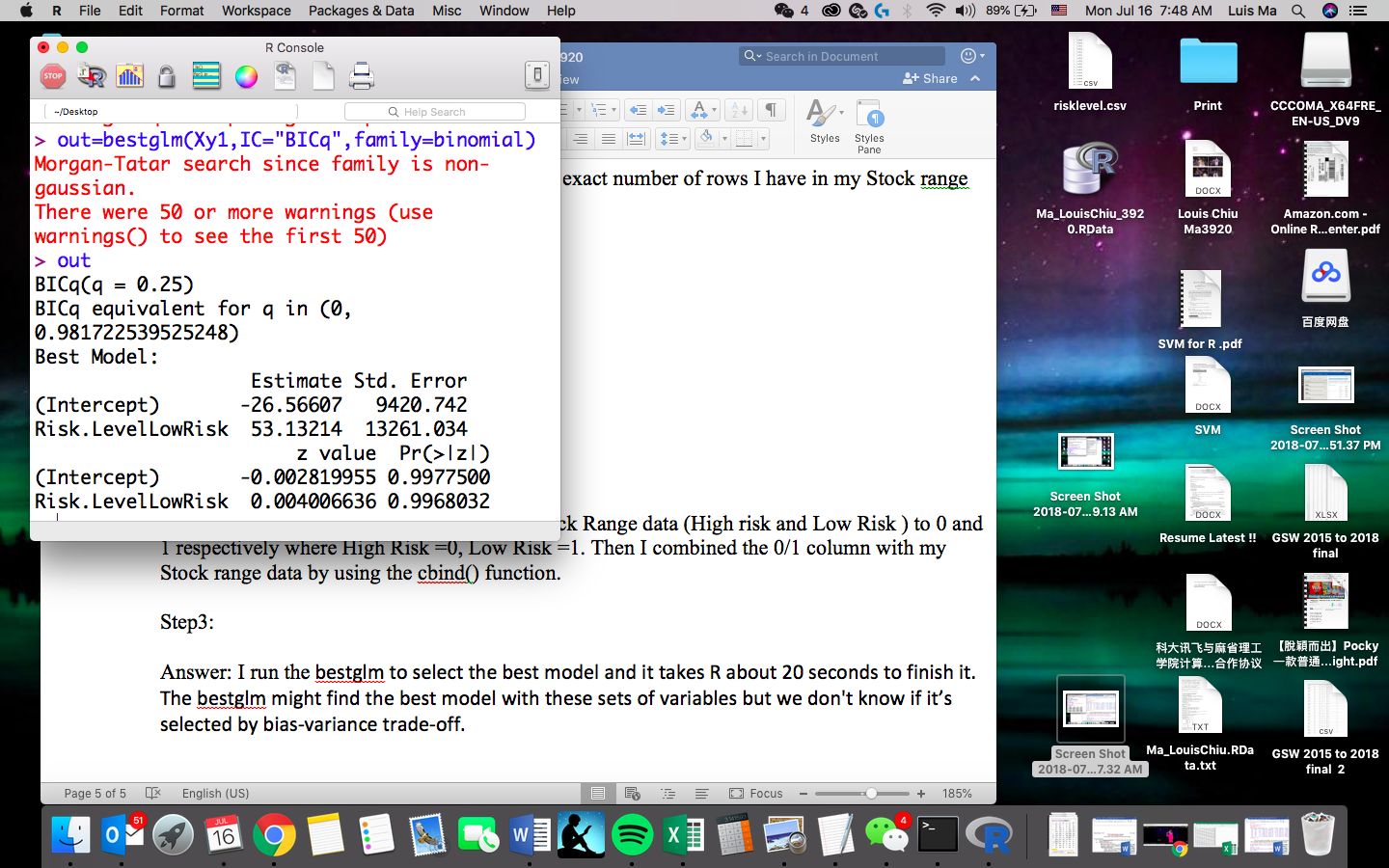
**Step 2 : Convert class to 1/0**

****

Answer: I convert the class (Risk Level)of Stock Range data (High risk and Low Risk ) to 0 and 1 respectively where High Risk =0, Low Risk =1. Then I combined the 0/1 column with my Stock range data by using the cbind() function.

**Step3: Apply bestglm**

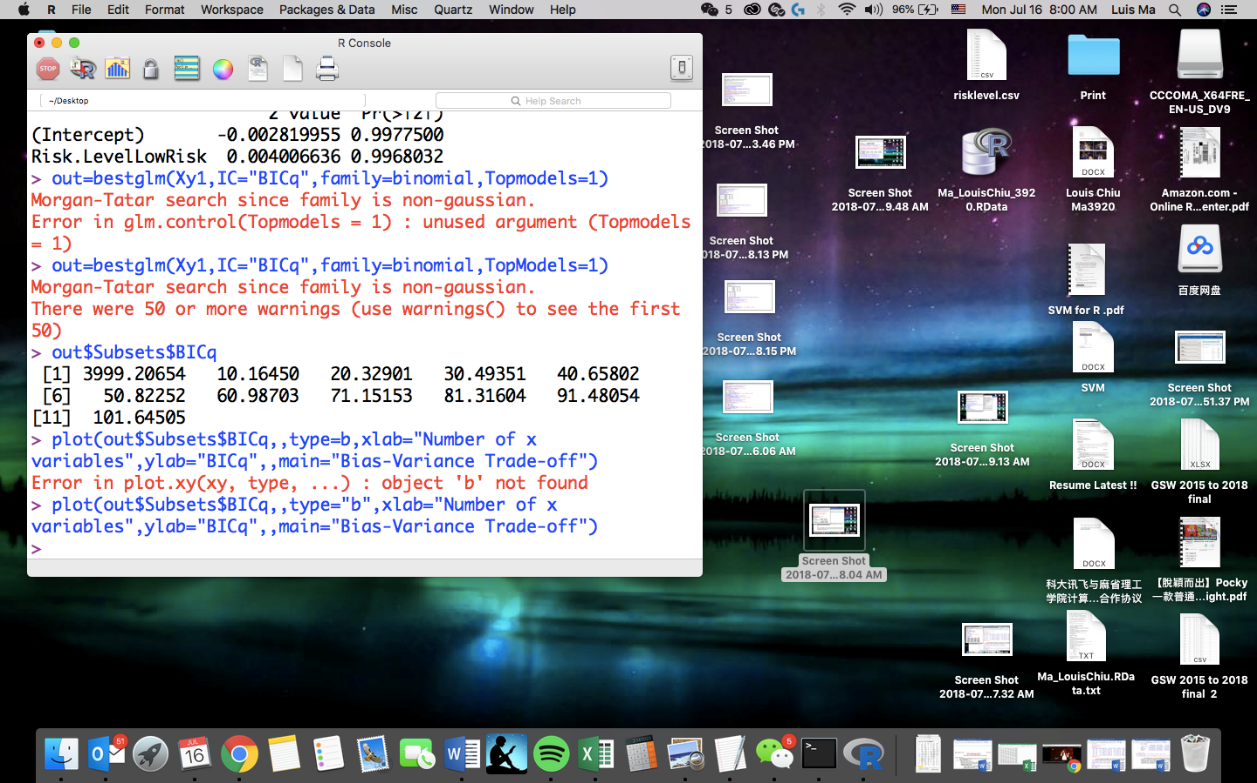




Answer: I run the bestglm to select the best model and it takes R about 20 seconds to finish it.

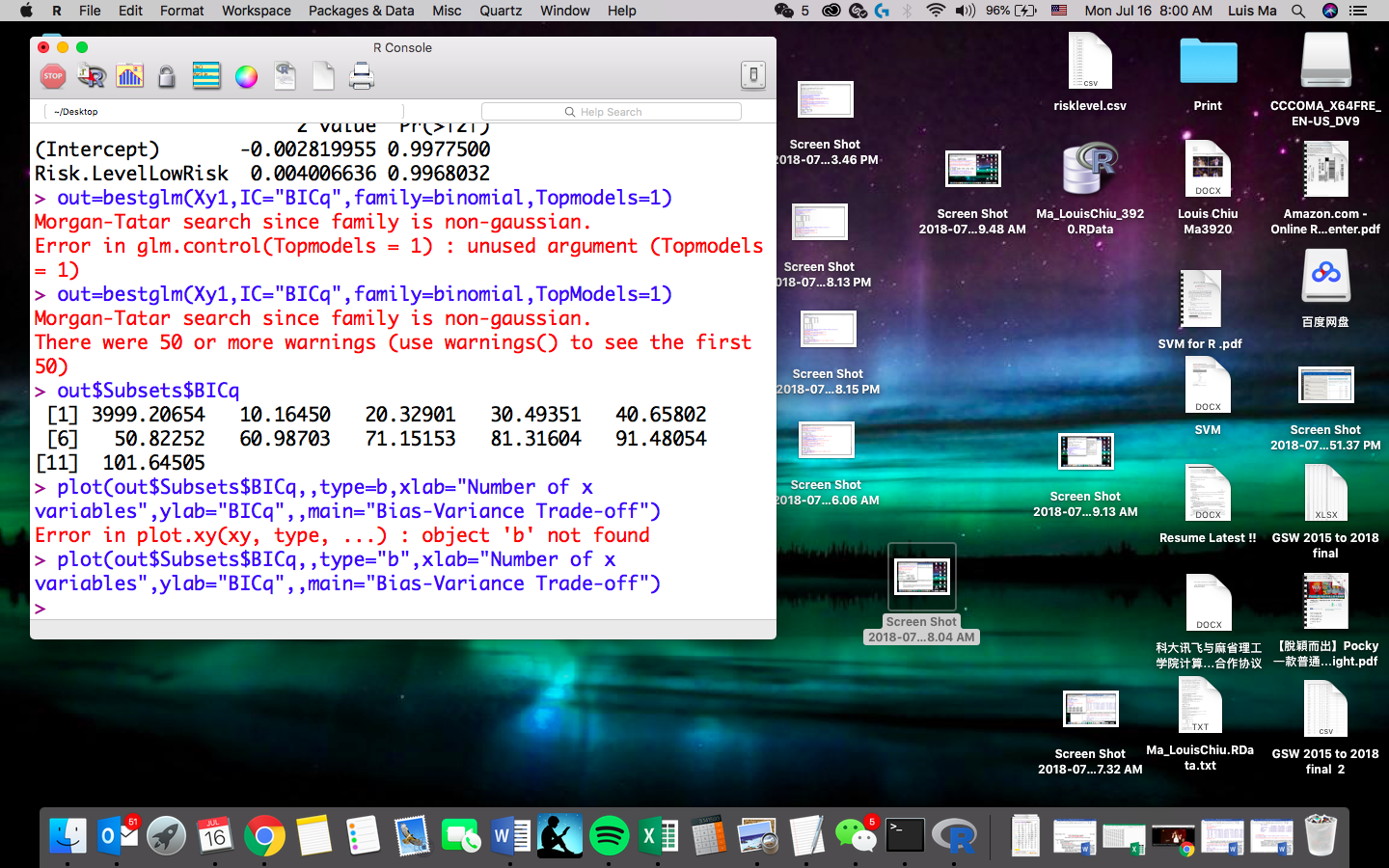
The bestglm might find the best model with these sets of variables but we don't know if it’s selected by bias-variance trade-off in the follow step.

**Step 4: Extract Bias-Variance Trade Off**



Answer: I’m using the TopModel=1 argument to find out the BICq value for the best model for each possible number of x-variables.

**Step 5: Graph visualization**



Answer: I then plot the graph to see the bias variance Trade-off among using increasing number of x-variable

Rplot.pdf

Observation: I can see the using 2 variables do perform better than that of using more in this case. In the other word, with 2 lags, I can actually predict or forecast the stock range better than that of using more lags based on the Bias Variance Trade-off with BICq.

**Code in R**

> IBMrange=read.csv("C:\\Users\\ \\Desktop\\LCMrange.csv")

> head(IBMrange)

Date IBMrange Lag1 Lag2

1 1/5/2006 1.65 1.17 2.74

2 1/6/2006 1.62 1.65 2.17

3 1/9/2006 0.87 1.62 2.65

4 1/10/2006 1.00 0.87 2.62

5 1/11/2006 1.41 1.00 1.87

6 1/12/2006 0.56 1.41 2.00

> tail(IBMrange)

Date IBMrange Lag1 Lag2

2534 1/29/2016 1.83 1.82 3.72

2535 2/1/2016 1.58 1.83 2.82

2536 2/2/2016 1.50 1.58 2.83

2537 2/3/2016 3.16 1.50 2.58

2538 2/4/2016 3.50 3.16 2.50

2539 2/5/2016 2.79 3.50 4.16

> sample(5,5)

[1] 5 3 2 4 1

> Shuffle=sample(2539,2539)

> IBMrange[Shuffle[1:6],]

Date IBMrange Lag1 Lag2

2420 8/17/2015 1.99 1.63 2.77

609 6/6/2008 3.40 1.84 3.05

1158 8/11/2010 1.23 1.72 2.94

2015 1/7/2014 3.97 2.06 3.05

239 12/14/2006 1.18 0.83 2.55

719 11/11/2008 4.04 5.36 3.46

> InSample=Shuffle[1:1200]

> OutSample=Shuffle[1201:2539]

> X.IBM=IBMrange[,3:4]

> Y.IBM=IBMrange[,2]

> median(Y.IBM)

[1] 1.98

> Y.IBM[Y.IBM>1.98]="HighRisk"

> Y.IBM[Y.IBM<=1.98]="LowRisk"

> Y.IBM[1:6]

[1] "LowRisk" "LowRisk" "LowRisk" "LowRisk" "LowRisk" "LowRisk"

> as.factor(Y.IBM[1:6])

[1] LowRisk LowRisk LowRisk LowRisk LowRisk LowRisk

Levels: LowRisk

> Y.IBM = as.factor(Y.IBM)

> TrainX.IBM=X.IBM[InSample,]

> TrainY.IBM=Y.IBM[InSample,]

Error in `[.default`(Y.IBM, InSample, ) : incorrect number of dimensions

> TrainY.IBM=Y.IBM[InSample]

> TestX.IBM=X.IBM[OutSample,]

> TestY.IBM=Y.IBM[OutSample]

> knn.pred = knn(TrainX.IBM, TestX.IBM, TrainY.IBM, 25)

> table(knn.pred,TestY.IBM)

TestY.IBM

knn.pred HighRisk LowRisk

HighRisk 486 193

LowRisk 213 447

> table.out = table(knn.pred,TestY.IBM)

> (table.out[1,1]+table.out[2,2])/sum(table.out)

[1] 0.6967886