BUS 41201 Big Data Midterm

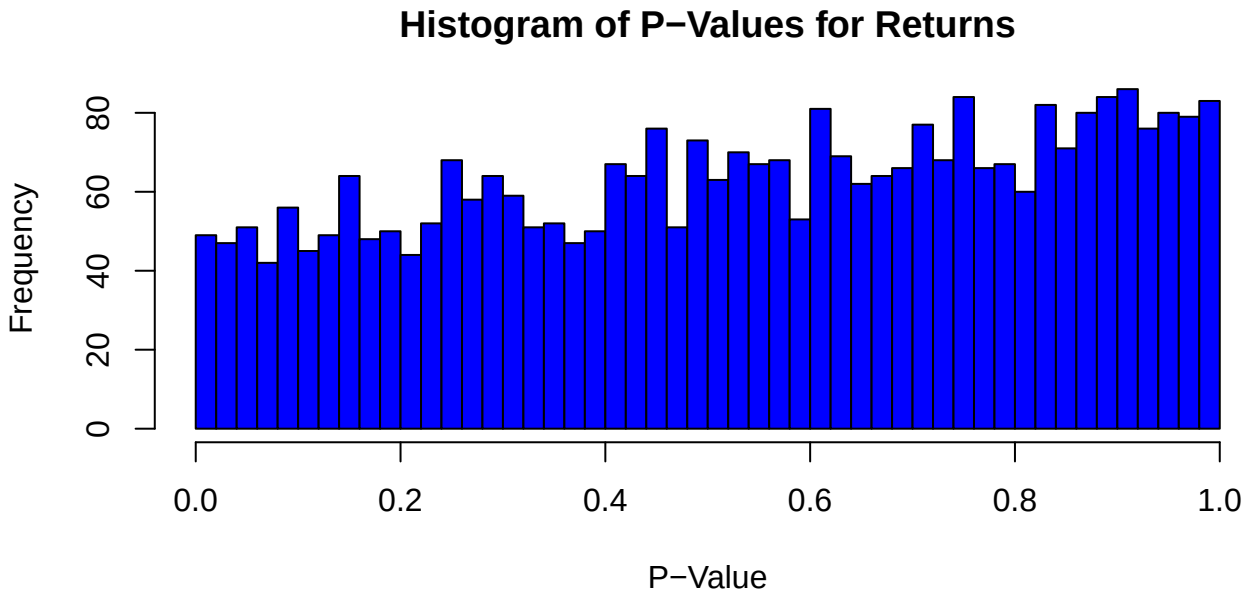
Shri Lekkala

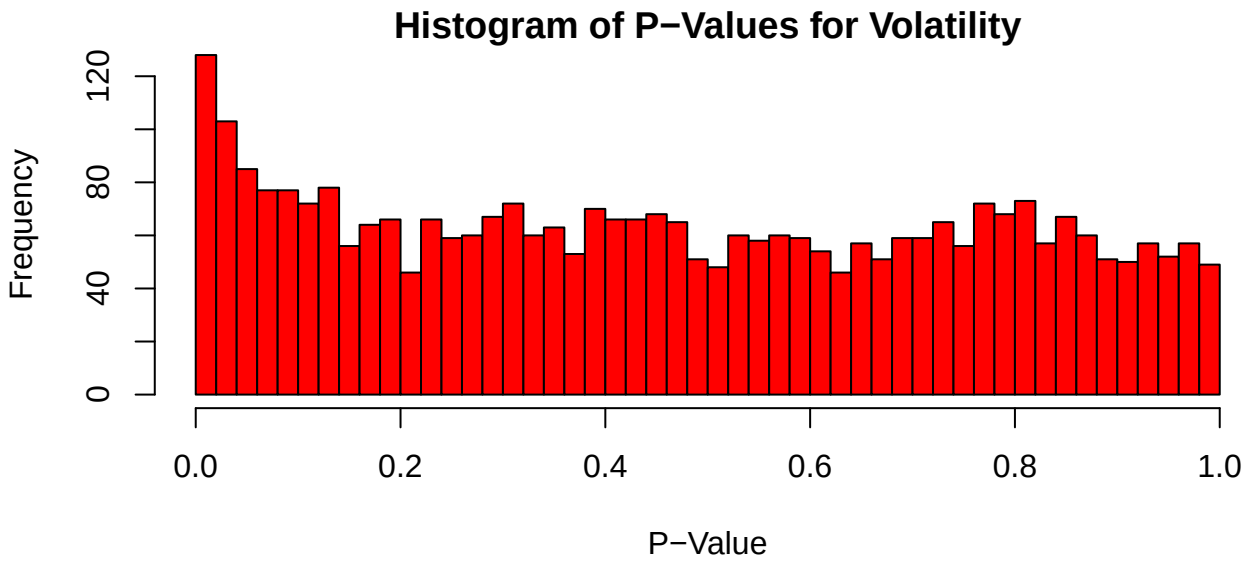
24 April 2024

Note: The full the code used in all the questions can be found in the appendix.

1 Marginal Significance Screening and False Discovery

Question (1.1)





The null hypothesis is a uniform distribution of p-values which would mean all p-values having the same frequency.

The histogram of p-values for returns, R, seems to be fairly uniform but seems to be somewhat skewed to the left, which suggests that there is likely to be little signal to predict the returns.

However for volatility, V, the p-values have a clear spike near 0 and seem to be skewed towards the right. The large amount of p-values near 0 suggests there might indeed be a signal for some words to predict volatility.

Question (1.2)

```
mgr_p_R = length(mrgpvals_R_ordered)
mrg_cutoff_R <- fdr_cut(mrgpvals_R_ordered, q=0.1)
mrg_cutoff_R
```

```
## [1] 1.026222e-05
```

```
mgr_p_V = length(mrgpvals_V_ordered)
mrg_cutoff_V <- fdr_cut(mrgpvals_V_ordered, q=0.1)
mrg_cutoff_V
```

```
## [1] 0.0003571024
```

At the 10% False Discovery rate, the p-value cutoff is:

- 1.026×10^{-5} (4.s.f) for R
- 3.571×10^{-4} (4.s.f) for V

```
# Number of significant words at alpha level 0.1
significant_R <- sum(mrgpvals_R < mrg_cutoff_R)
significant_R
```

```
## [1] 0
```

```
significant_V <- sum(mrgpvals_V < mrg_cutoff_V)
significant_V
```

```
## [1] 11
```

At this FDR level, there are 0 words are significant for R and 11 words significant for V.

The main advantages of using FDR analysis for word selection is the ability to handle large datasets and do computations in parallel. So as an initial stage for analyzing which words are important, this method is computationally efficient compared to others. Further, as we are explicitly controlling the expected number of false discoveries, this method allows for a more reliable selection of significant results.

However, the FDR method relies on the assumption that each test is independent. This is not necessarily true for words in headlines as words can occur simultaneously. Further, we completely disregard the structure of groups of words / phrases by treating each word independently. Another disadvantage is that the FDR analysis only selects words based on the magnitude of the association but not their direction, so we cannot use this to select only words with positive or negative association.

Question (1.3)

Just marking the 20 smallest p-values without using any testing correction like FDR lacks statistical control over the expected number of false discoveries. So we cannot estimate the expected number of false discoveries. However, if we used the discoveries using a 10% FDR from above, then we can expect 10% of the 11 words, i.e. 1.1 of them to be false discoveries.

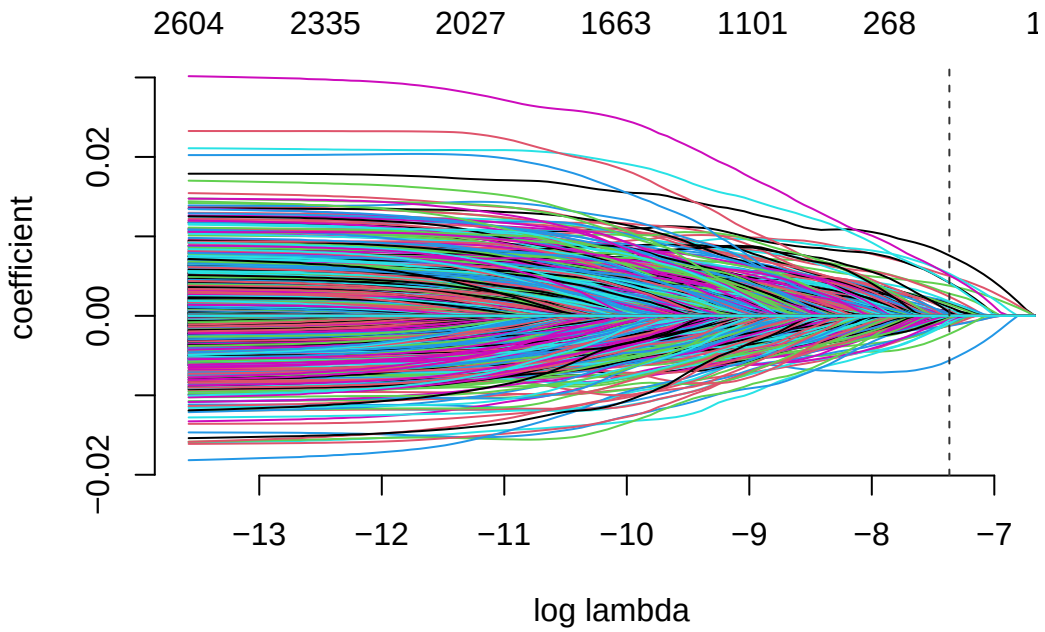
However, it is unlikely that the p-values are independent as words in language are usually correlated. So some words are likely to appear next to each other, and the presence of one word could mean that another word is more likely to appear. This is true with news headlines which often have common or repeated phrases, which means that some words appear frequently with other words.

Thus, taking into account the lack of independence, we may assume that the expected number of false discoveries is even higher due to dependencies between words not being taken into account. So in conclusion, a more rigorous statistical method should be applied to find significant words rather than simply marking the smallest n as significant.

2 LASSO Variable Selection and Bootstrapping

Question (2.1)

```
# First analyze returns
lasso1 <- gamlr(spm, y=R, lambda.min.ratio=1e-3)
plot(lasso1)
```



The top 10 most predictive words from this model are computed below:

```
betas <- drop(coef(lasso1)) # AICc default selection
non_zero_betas = betas[betas!=0][-1] # Exclude intercept and filter out zero betas

# Choose 10 most predictive words in this model
o<-order(non_zero_betas,decreasing=TRUE)
kable(non_zero_betas[o[1:10]])
```

	x
damn	0.0075899
theyll	0.0053115
elect	0.0051119
jamaican	0.0050146
kiss	0.0041769
trawler	0.0041643
paul	0.0038002
rift	0.0027937
grip	0.0024081
cctv	0.0018005

```r
# Pick lambda using the AICc
lambda_1 = lasso1$lambda[which.min(AICc(lasso1))]
paste0("The AICc pick for lambda is: ", lambda_1[1])
```

```
## [1] "The AICc pick for lambda is: 0.000631890583514516"
```

```r
# Compute the in-sample R2
dev <- lasso1$deviance[which.min(AICc(lasso1))] # deviance #of the AICc selected model
dev0<- lasso1$deviance[1] # null deviance
paste0("The in-sample $R^2$ is: ", 1-dev/dev0)
```

```
## [1] "The in-sample $R^2$ is: 0.07200122438874"
```

The in-sample R^2 is 0.07200122 which suggests there is not much signal. This means that approximately 7.2% of the variance in returns R is explained by the LASSO model based on word frequencies from the headlines. Having a low R^2 score suggests that is not sufficient evidence to conclude that headlines predict returns.

Question (2.2)

```r
# **** LASSO Analysis of volatility **** #
lasso2 <- gamlr(spm, y=V, lambda.min.ratio=1e-3)
betas2 <- drop(coef(lasso2)) # AICc default selection
non_zero_betas2 = betas2[betas2!=0][-1] # Exclude intercept and filter out zero betas

# Choose 10 most predictive words in this model
o <- order(non_zero_betas2,decreasing=TRUE)
kable(non_zero_betas2[o[1:10]])
```

	x
republican	0.1954297
chunk	0.1632192
fusion	0.1455132
govern	0.1123943
shed	0.0870666
august	0.0866411
bailout	0.0786730
barrel	0.0698835
hussein	0.0682012
medvedev	0.0658347

```r
# Pick lambda using the AICc
lambda_2 = lasso2$lambda[which.min(AICc(lasso2))]
paste0("The AICc pick for lambda is: ", lambda_2[1])
```

```
## [1] "The AICc pick for lambda is: 0.0260163567306549"
```

```r
# Compute the in-sample R2
dev_2 <- lasso2$deviance[which.min(AICc(lasso2))] # deviance #of the AICc selected model
dev0_2<- lasso2$deviance[1] # null deviance
paste0("The in-sample $R^2$ is: ", 1-dev_2/dev0_2)
```

```
## [1] "The in-sample $R^2$ is: 0.161611573416107"
```

So the in-sample R^2 for this model to predict volatility is greater than the previous one for returns, which suggests a stronger signal for prediction.

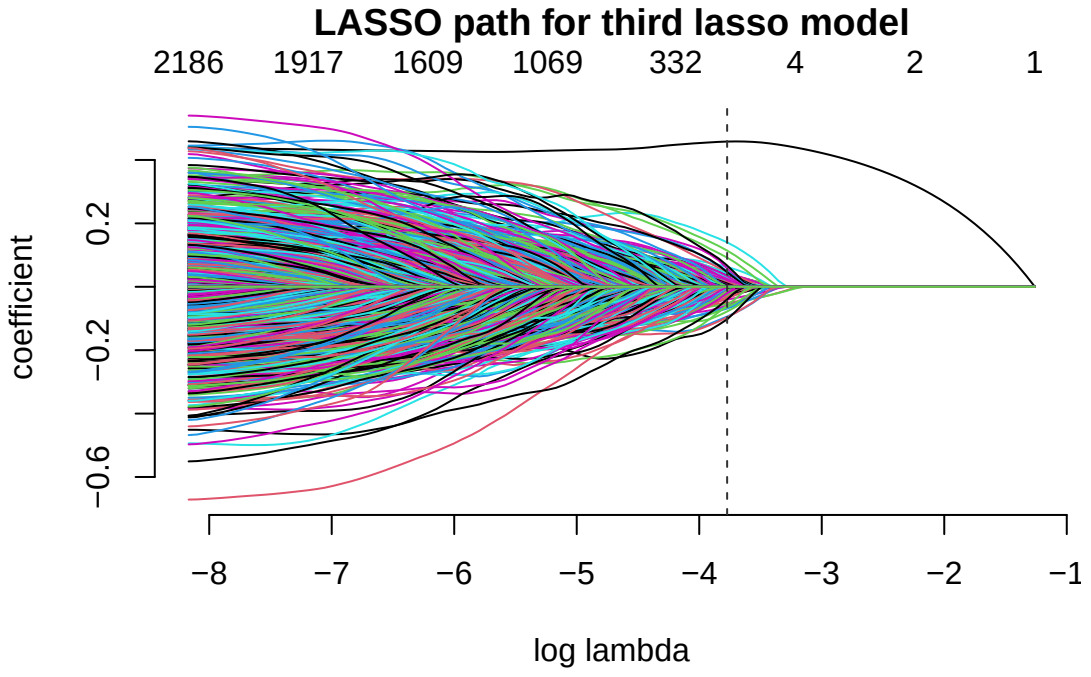
Now we fit a LASSO model for: $V_t = a + bV_{t-1} + x_t'\beta + \epsilon_t$

```r
# Pick lambda using the AICc
lambda_3 = lasso3$lambda[which.min(AICc(lasso3))]

# Compute the in-sample R2
dev_3 <- lasso3$deviance[which.min(AICc(lasso3))] # deviance #of the AICc selected model
dev0_3 <- lasso3$deviance[1] # null deviance
paste0("The in-sample $R^2$ is: ", 1-dev_3/dev0_3)
```

```
## [1] "The in-sample $R^2$ is: 0.331584990647596"
```

So the in-sample R^2 of the third model is much greater than the previous 2 models, and suggests an even stronger signal, so adding the previous day's volatility increased our predictive power.



From the plot of the coefficients vs lambda, we observe that the model iteratively selects coefficients of different words to become 0 as the value of lambda increases. The dotted vertical line corresponds to the log value of the lambda selected by the AICc.

The coefficients all tend to 0 fairly sequentially as lambda decreases, however there is one coefficient that remains non-zero as lambda increases, and only drops to 0 after a significant rise in lambda. This is likely to be the coefficient for the previous day's volatility (V_{t-1}).

```
# Choose top 10 strongest coefficients in this model
betas3 <- drop(coef(lasso3)) # AICc default selection
non_zero_betas3 = betas3[betas3!=0][-1] # Exclude intercept and filter out zero betas

# Choose 10 most predictive words in this model
o <- order(non_zero_betas3,decreasing=TRUE)
kable(non_zero_betas3[o[1:10]])
```

	x
previous	0.4580944
shed	0.1401080
fusion	0.1141525
republican	0.0920512
pioneer	0.0670934
lowest	0.0629167
chunk	0.0581594
ton	0.0491526
william	0.0463384
medvedev	0.0450435

The top 10 strongest coefficients are listed above.

```
# coefficients
coef_terrorist = betas3[names(betas3)=="terrorist"]
paste0("The coefficient of the word terrorist is: ", coef_terrorist)
```

```
## [1] "The coefficient of the word terrorist is: 0.0178389418998624"
```

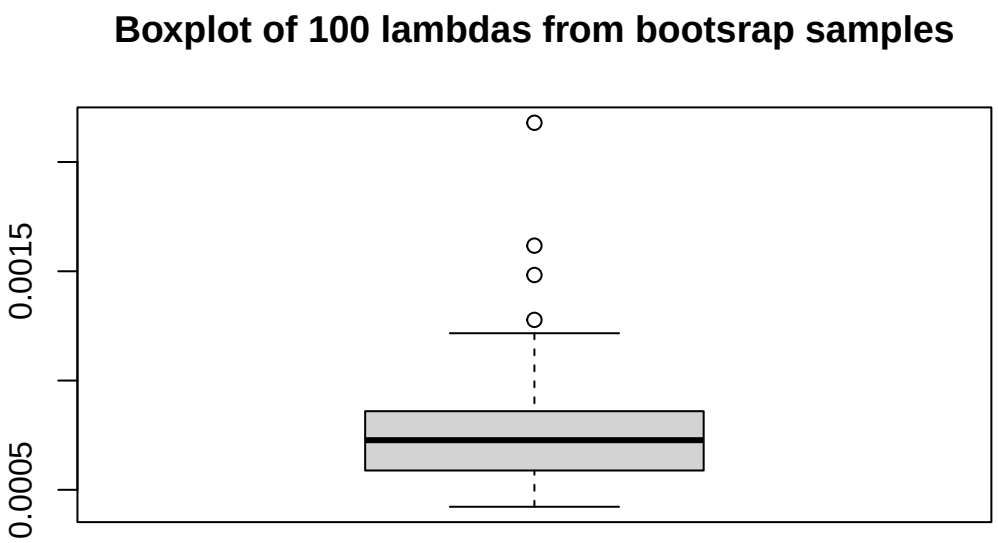
```
# V_{t-1}
coef_prev_v = betas3[names(betas3)=="previous"][1]
paste0("The coefficient of $V_{t-1}$ is: ", coef_prev_v)
```

```
## [1] "The coefficient of $V_{t-1}$ is: 0.458094444835972"
```

The coefficient of the word "terrorist" suggests a small but positive association between the word appearing in headlines and market volatility that day. So the appearance of the word "terrorist" suggests that the market will increase in volatility by ≈ 0.0178 units keeping everything else constant.

The coefficient of the previous day's volatility is positive and large suggesting that the previous day's volatility has a strong positive effect on the current day's volatility. That is, a increase in the yesterday's volatility by 1 unit leads to an expected increase in today's volatility by ≈ 0.458 units, keeping other factors fixed. So V_{t-1} is a critical component in forecasting V_t .

Question (2.3)



The summary statistics for the bootstrap samples are:

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 0.0004363 0.0006346 0.0007588 0.0007849 0.0009320 0.0013180
```

```
# Original estimate of lambda
paste0("The original estimated lambda from 2.2 is: ", lambda_3)
```

```
## [1] "The original estimated lambda from 2.2 is: 0.0230003726568119"
```

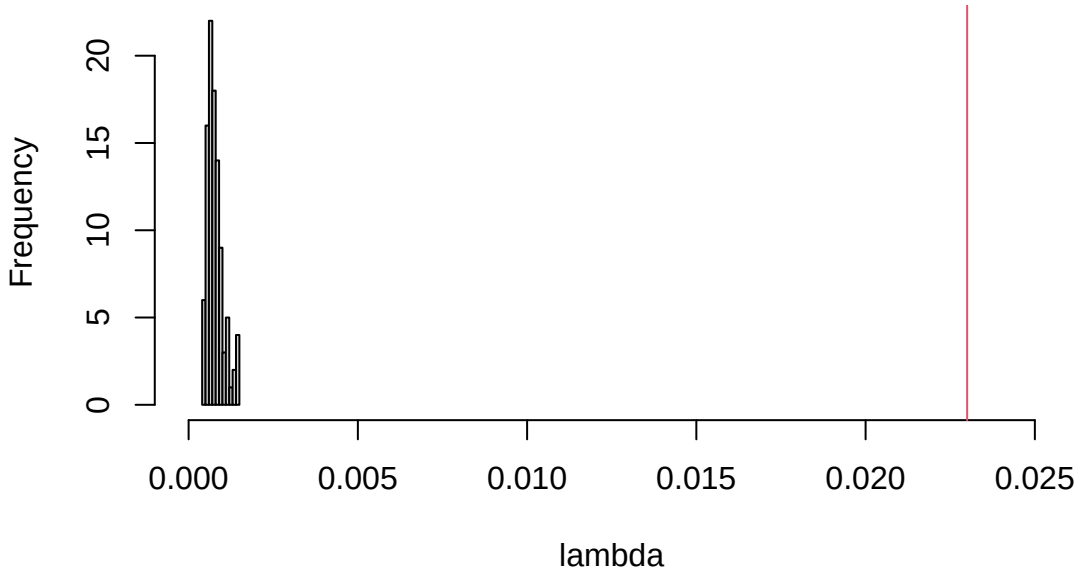
```
# Compute Standard Error
se = sqrt(mean((lambda_samp - lambda_3)**2))
paste0("The standard error for the selected lambda is: ", se)
```

```
## [1] "The standard error for the selected lambda is: 0.0222163308069742"
```

```
# Compute the 95% confidence interval
lower = lambda_3 - 1.96 * se
upper = lambda_3 + 1.96 * se

print(paste0("The 95% CI for the selected lambda is: [", lower, ",", upper, "]"))
```

```
## [1] "The 95% CI for the selected lambda is: [-0.0205436357248576,0.0665443810384815]"
```



The red line on the histogram indicates the estimated lambda from (2.2). This shows that our our full sample AICc lambda is very far from our bootstrap sampling distribution.
This suggests that there is high variability in the estimator and this leads us to question the robustness and reliability of the original estimate.

3 High-dimensional Controls and Double LASSO

Question (3.1)

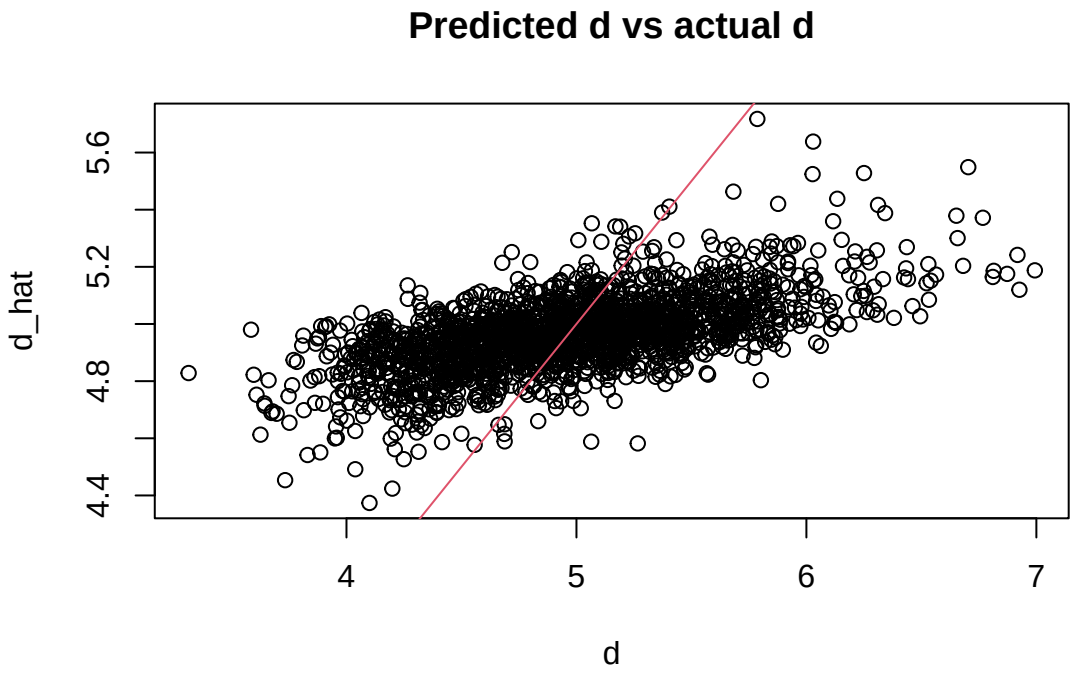
```
summary(glm(V~d))$coef
```

```
##               Estimate Std. Error  t value      Pr(>|t|)
## (Intercept) 2.4216818 0.09619961 25.17351 1.346083e-121
## d           0.5119538 0.01926182 26.57868 2.010805e-133
```

The coefficient of d is 0.51195 is positive and significant (p-value $<$ 2e-16) which suggests a strong positive correlation between d and V. That is, for a one-unit increase in yesterday's volatility, we can expect an increase of ≈ 0.512 units in today's volatility.

```
# LASSO model to predict d from x
treat <- gamlr(spm, d, lambda.min.ratio=1e-4)

# Isolate d_hat (the part of treatment that we can predict with x's)
d_hat = predict(treat, spm)
```




```
# In-sample R2
print(paste0("R2: ", cor(d, drop(d_hat))^2))
```

```
## [1] "R2: 0.364826258601986"
```

From the graph we observe that the predictions are fairly close to the actual values, but the slope of the trend differs from what we expect.

Also the in-sample R^2 suggests that $\approx$ 36.5 % of the variance in d is explained by x (the words in the headlines).

As the words explain more than 1/3 of the amount of variance in V_{t-1}, we can expect a moderate degree of confounding effect that should be controlled.

However, there is still $\approx 63.5\%$ of unexplained variance in V_{t-1} which suggests that there is significant information in d that is independent of x.

Question (3.2)

```
# Second Stage LASSO
causal <-  gamlr(cbind(d, d_hat, spm), V, free=2)
print(paste0("The coefficient of d from 2-stage LASSO is: ", coef(causal)["d",]))
```

```
## [1] "The coefficient of d from 2-stage LASSO is: 0.360282069419024"
```

```
# Naive LASSO
naive <- gamlr(cbind(d, spm), V)
print(paste0("The coefficient of d from naive LASSO is: ", coef(naive)["d",]))
```

```
## [1] "The coefficient of d from naive LASSO is: 0.457421796653875"
```

The effect of V_{t-1} from the causal LASSO is ≈ 0.36 which is fairly large, thus suggests that there is evidence that the previous day's volatility does have a moderate effect on the current day's volatility. So after controlling for confounding effects of the headline words, a 1 unit increase in yesterday's volatility leads to an increase of approximately 0.36 units in today's volatility.

However we also observe that there is an effect when running naive LASSO, and in fact it is greater in magnitude (≈ 0.46). This difference in magnitude might be attributed to the confounding influence of the words in the headlines.

Question (3.3)

By implementing a double LASSO, we did manage to effectively control for the observed confounders x (the words in the headlines that might influence both V_t and V_{t-1}), and the coefficient we obtained was not insignificant.

However, this is just one part of the picture, as we have not controlled for all confounders, namely the unobserved ones that are not included in our datasets. These may still influence both V_t and V_{t-1} so we cannot conclude with certainty that there is a causal effect.

Further, as with all statistical analyses, we need to make sure the model is correctly specified, and statistical assumptions are met. As discussed in question 1, the independence of words in the headlines may not be a reasonable assumption to make. And as we are dealing with financial data, it is unclear whether the assumption of normality for error terms still holds, which is necessary to interpret the model output.

Finally, in order reach a stronger conclusion about causality, we should ideally implement further analyses that can either strengthen or disprove our claim, such as the use of instrumental variables, or more advanced LASSO methods.

Thus although we cannot claim for certain the effect is causal, we can claim that there is a statistically significant association and so our double LASSO model suggests that there could be a potential causal effect.

Bonus Freestyle Analysis

We will try to perform sentiment analysis of the news headlines to try and correlate it with stock market volatility.

(Note the full code can be found in the appendix.)

```
# Calculate daily sentiment scores
daily_sentiment <- word_data_sentiment %>%
  group_by(day) %>%
  summarize(sentiment_score = sum(frequency * score))

# Merge sentiment and volatility data
volatility_data <- data.frame(day = as.numeric(rownames(daily_sentiment)), volatility = V)
analysis_data <- merge(daily_sentiment, volatility_data, by = "day")

summary(analysis_data)
```

```
##      day            sentiment_score     volatility
##  Length:1988        Min.   :-20.000   Min.   :3.313
##  Class :character   1st Qu.: -7.000   1st Qu.:4.564
##  Mode  :character   Median : -4.000   Median :4.948
##                     Mean   : -4.455   Mean   :4.963
##                     3rd Qu.: -2.000   3rd Qu.:5.320
##                     Max.   :  9.000   Max.   :6.993
```

Now we compute the correlation between the daily sentiment score and the volatility:

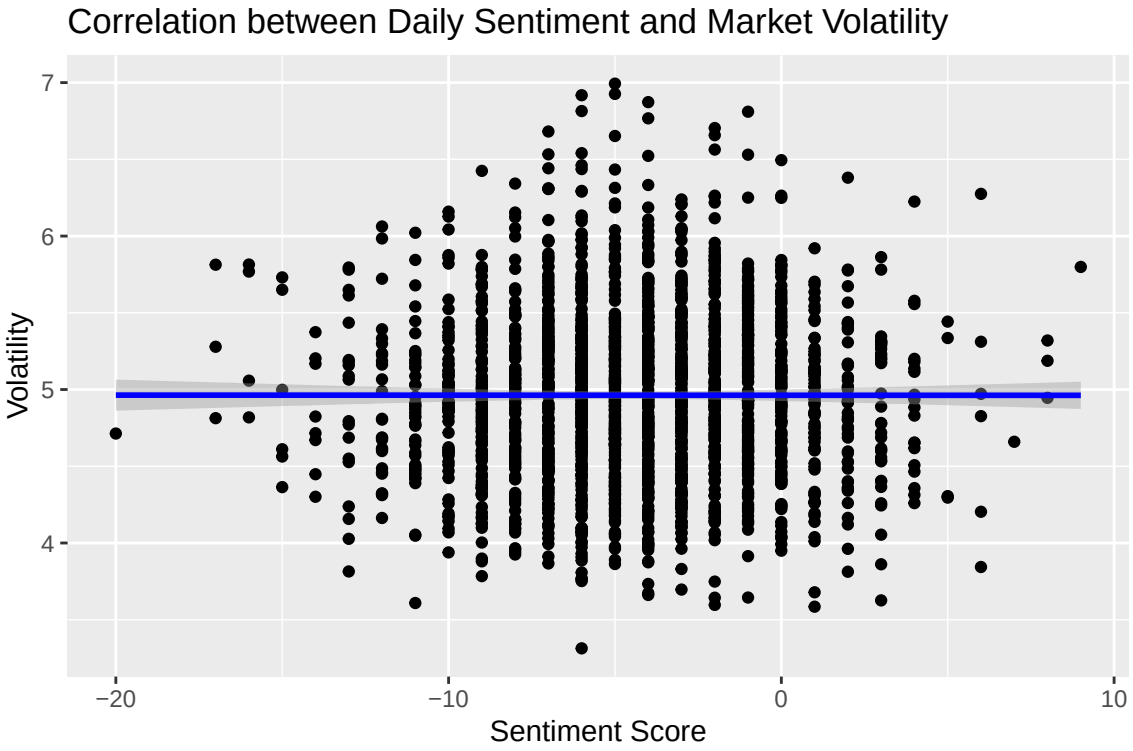
```
correlation_result <- cor(analysis_data$sentiment_score, analysis_data$volatility)

print(paste0("Correlation between sentiment and volatility:", correlation_result))
```

```
## [1] "Correlation between sentiment and volatility:-0.000312410192881722"
```

So we observe that the correlation is very close to 0, which suggests that there is virtually no linear relationship between the daily sentiment from the news headlines and the market volatility that day.

To see if we can identify any trends we can look at the plot:



So although there is little evidence to suggest any linear relationship. However there could be a possibility for non-linear or more complex relationships between the sentiment of the daily news and volatility.

Further, there is noticeable variation in the range of volatility for each sentiment score. It seems that the closer the daily sentiment is to being between -10 and 0, the larger the variation, however having a more extreme sentiment score seem to have less variability. But we cannot make such a conclusion just from looking at the graph alone, and this may be explained due to other the lack of days with extreme sentiment scores in our data set.

Overall, from our analysis we cannot conclude that their is any clear impact of the daily sentiment on market volatility. However, in order to refine the analysis, we could try and take a more nuanced approach by considering sentiment thresholds and investigating whether extreme sentiments have a different effect compared to moderate ones.

It should be noted that the sentiment dictionary is limited, and does not include general topics, names, or events which often have a sentiment attached to them in practice. A further approach to improve our sentiment analysis might be to categorize news based on content using tools that account for context and intensity.

Appendix

```
########################################
# Setup
########################################

knitr::opts_chunk$set(
    echo = FALSE,
    fig.height = 4,
    fig.width = 6,
    warning = FALSE,
    cache = TRUE,
    digits = 3,
    width = 48
)


library(readtext)
library(SnowballC)
library(tidytext)
library(gamlr)
library(parallel)
library(knitr)
# Reddit news data
data = read.csv("RedditNews.csv", header = FALSE, skip = 1)
data[,2:3] = data[,1:2]
data[,1] = paste0("RedditNews_",rownames(data))
date<-data[2] # this is the day of the news

subset<-date=="7/1/16" # let's take a look at news headlines on 7/1/16
# data[subset,3] # we have 24 news headlines
# Read the DJIA data
dj<-read.csv("DJIA.csv")
head(dj) # Open price, highest, lowest and close price
ndays<-nrow(dj) # 1989 days
# Read the words
words<-read.csv("WordsFinal.csv",header=F)
words<-words[,1]
head(words)
length(words)
# Read the word-day pairings
doc_word<-read.table("WordFreqFinal.csv",header=F)

# Create a sparse matrix
spm<-sparseMatrix(
        i=doc_word[,1],
        j=doc_word[,2],
        x=doc_word[,3],
        dimnames=list(id=1:ndays,words=words))
dim(spm)

# We select only words at occur at least 5 times
cols<-apply(spm,2,sum)
```

```r
index<-apply(spm,2,sum)>5
spm<-spm[,index]

# and words that do not occur every day
index<-apply(spm,2,sum)<ndays
spm<-spm[,index]

dim(spm) # we end up with 3183 words

##########################################
# (1.1)
##########################################

#  *** FDR *** analysis
spm<-spm[-ndays,]
time<-dj[-ndays,1]

# Take returns
par(mfrow=c(1,2))
R<-(dj[-ndays,7]-dj[-1,7])/dj[-1,7]
plot(R~as.Date(time),type="l")
title("Returns against time")

# Take the log of the maximal spread
V<-log(dj[-ndays,3]-dj[-ndays,4])
plot(V~as.Date(time),type="l")
title("Volatility against time")
# FDR: we want to pick a few words that correlate with the outcomes (returns and volatility)

# create a dense matrix of word presence
P <- as.data.frame(as.matrix(spm>0))

# we will practice parallel computing now
margreg <- function(x){
    fit <- lm(Outcome~x)
    sf <- summary(fit)
    return(sf$coef[2,4])
}
# **** Analysis for Returns ****
cl <- makeCluster(detectCores())

Outcome<-R
clusterExport(cl,"Outcome")

# run the regressions in parallel
mrgpvals_R <- unlist(parLapply(cl,P,margreg))

stopCluster(cl)

# **** Analysis for Volatility ****
cl <- makeCluster(detectCores())

Outcome <- V
```

```r
clusterExport(cl,"Outcome")

# run the regressions in parallel
mrgpvals_V <- unlist(parLapply(cl,P,margreg))

stopCluster(cl)
# Combine the p-values into a data frame for easy plotting
pvals <- data.frame(Returns = mrgpvals_R, Volatility = mrgpvals_V)

# Plotting the p-values
# Plotting the histograms of p-values

# Histogram for Returns
hist(mrgpvals_R, breaks = 50, main = NULL, xlab = "P-Value", ylab = "Frequency", col="blue")
title("Histogram of P-Values for Returns", line=1)
# Histogram for Volatility
hist(mrgpvals_V, breaks = 50, main = NULL, xlab = "P-Value", ylab = "Frequency", col="red")
title("Histogram of P-Values for Volatility", line=0.1)


##########################################
# (1.2)
##########################################

# To find the p-value cut off we first order the p values
mrgpvals_R_ordered <- mrgpvals_R[order(mrgpvals_R, decreasing=F)]
mrgpvals_V_ordered <- mrgpvals_V[order(mrgpvals_V, decreasing=F)]

source("fdr.R")
mgr_p_R = length(mrgpvals_R_ordered)
mrg_cutoff_R <- fdr_cut(mrgpvals_R_ordered, q=0.1)
mrg_cutoff_R

mgr_p_V = length(mrgpvals_V_ordered)
mrg_cutoff_V <- fdr_cut(mrgpvals_V_ordered, q=0.1)
mrg_cutoff_V
# Number of significant words at alpha level 0.1
significant_R <- sum(mrgpvals_R < mrg_cutoff_R)
significant_R
significant_V <- sum(mrgpvals_V < mrg_cutoff_V)
significant_V

##########################################
# (2.1)
##########################################

# First analyze returns
lasso1 <- gamlr(spm, y=R, lambda.min.ratio=1e-3)
plot(lasso1)
betas <- drop(coef(lasso1)) # AICc default selection
non_zero_betas = betas[betas!=0][-1] # Exclude intercept and filter out zero betas

# Choose 10 most predictive words in this model
```

```
o<-order(non_zero_betas,decreasing=TRUE)
kable(non_zero_betas[o[1:10]])
# Pick lambda using the AICc
lambda_1 = lasso1$lambda[which.min(AICc(lasso1))]
paste0("The AICc pick for lambda is: ", lambda_1[1])


# Compute the in-sample R2
dev <- lasso1$deviance[which.min(AICc(lasso1))] # deviance #of the AICc selected model
dev0<- lasso1$deviance[1] # null deviance
paste0("The in-sample $R^2$ is: ", 1-dev/dev0)

########################################
# (2.2)
########################################

# **** LASSO Analysis of volatility **** #
lasso2 <- gamlr(spm, y=V, lambda.min.ratio=1e-3)
betas2 <- drop(coef(lasso2)) # AICc default selection
non_zero_betas2 = betas2[betas2!=0][-1] # Exclude intercept and filter out zero betas

# Choose 10 most predictive words in this model
o <- order(non_zero_betas2,decreasing=TRUE)
kable(non_zero_betas2[o[1:10]])
# Pick lambda using the AICc
lambda_2 = lasso2$lambda[which.min(AICc(lasso2))]
paste0("The AICc pick for lambda is: ", lambda_2[1])

# Compute the in-sample R2
dev_2 <- lasso2$deviance[which.min(AICc(lasso2))] # deviance #of the AICc selected model
dev0_2<- lasso2$deviance[1] # null deviance
paste0("The in-sample $R^2$ is: ", 1-dev_2/dev0_2)
# predict future volatility from past volatility
Previous <- log(dj[-1,3]-dj[-1,4]) # remove the last return
spm2 <- cbind(Previous,spm) # add the previous return to the model matrix
colnames(spm2)[1]<-"previous" # the first column is the previous volatility
lasso3 <- gamlr(spm2, y=V, lambda.min.ratio=1e-3)
# Pick lambda using the AICc
lambda_3 = lasso3$lambda[which.min(AICc(lasso3))]

# Compute the in-sample R2
dev_3 <- lasso3$deviance[which.min(AICc(lasso3))] # deviance #of the AICc selected model
dev0_3 <- lasso3$deviance[1] # null deviance
paste0("The in-sample $R^2$ is: ", 1-dev_3/dev0_3)
plot(lasso3)
title("LASSO path for third lasso model")
# Choose top 10 strongest coefficients in this model
betas3 <- drop(coef(lasso3)) # AICc default selection
non_zero_betas3 = betas3[betas3!=0][-1] # Exclude intercept and filter out zero betas

# Choose 10 most predictive words in this model
o <- order(non_zero_betas3,decreasing=TRUE)
kable(non_zero_betas3[o[1:10]])
```

```r
# coefficients
coef_terrorist = betas3[names(betas3)=="terrorist"]
paste0("The coefficient of the word terrorist is: ", coef_terrorist)

# V_{t-1}
coef_prev_v = betas3[names(betas3)=="previous"][1]
paste0("The coefficient of $V_{t-1}$ is: ", coef_prev_v)

##########################################
# (2.3)
##########################################

# Bootstrap to obtain s.e. of 1.s.e. chosen lambda

# We apply bootstrap to approximate the sampling distribution of lambda selected by AICc

# export the data to the clusters
cl <- makeCluster(detectCores())

Outcome<-V
clusterExport(cl,"spm2")
clusterExport(cl,"V")

# run 100 bootstrap resample fits
boot_function <- function(ib){
    require(gamlr)
    fit <- gamlr(spm2[ib,],y=V[ib], lambda.min.ratio=1e-3)
    fit$lambda[which.min(AICc(fit))]
}

boots <- 100
n <- nrow(spm2)
resamp <- as.data.frame(
            matrix(sample(1:n,boots*n,replace=TRUE),
            ncol=boots))

lambda_samp <- unlist(parLapply(cl,resamp,boot_function))
stopCluster(cl)

boxplot(lambda_samp)
title("Boxplot of 100 lambdas from bootsrap samples")
summary(lambda_samp)
# Original estimate of lambda
paste0("The original estimated lambda from 2.2 is: ", lambda_3)

# Compute Standard Error
se = sqrt(mean((lambda_samp - lambda_3)**2))
paste0("The standard error for the selected lambda is: ", se)
# Compute the 95% confidence interval
lower = lambda_3 - 1.96 * se
upper = lambda_3 + 1.96 * se

print(paste0("The 95% CI for the selected lambda is: [", lower, ",", upper, "]"))
```

```r
hist(lambda_samp, xlim=c(0, 0.025), main="Histogram of bootstrapped lambdas", xlab="lambda")
abline(v=lambda_3, col=2)

##########################################
# (3.1)
##########################################

# High-dimensional Covariate Adjustment
d <- Previous # this is the treatment
# marginal effect of past on present volatility
summary(glm(V~d))
summary(glm(V~d))$coef
# LASSO model to predict d from x
treat <- gamlr(spm, d, lambda.min.ratio=1e-4)

# Isolate d_hat (the part of treatment that we can predict with x's)
d_hat = predict(treat, spm)
# Plot d_hat against d
plot(d, drop(d_hat), ylab="d_hat")
abline(a=0, b=1, col=2)
title("Predicted d vs actual d")
# In-sample R2
print(paste0("R2: ", cor(d, drop(d_hat))^2))

##########################################
# (3.2)
##########################################

# Second Stage LASSO
causal <-  gamlr(cbind(d, d_hat, spm), V, free=2)
print(paste0("The coefficient of d from 2-stage LASSO is: ", coef(causal)["d",]))
# Naive LASSO
naive <- gamlr(cbind(d, spm), V)
print(paste0("The coefficient of d from naive LASSO is: ", coef(naive)["d",]))

##########################################
# (BONUS)
##########################################

library(Matrix)
library(dplyr)
library(tidyr)
library(ggplot2)

# Obtain sentiment dictionary
sentiments <- get_sentiments("bing")
# Assign scores to sentiments
sentiments <- sentiments |> mutate(score = if_else(sentiment =="positive", 1, -1))

# Convert the sparse matrix to a data frame
word_data <- as.data.frame(as.matrix(spm), stringsAsFactors = FALSE)
word_data$word <- rownames(spm)
word_data_long <- reshape2::melt(word_data, id.vars = "word")
```

```r
names(word_data_long) <- c("day", "word", "frequency")
word_data_long <- word_data_long |> filter(frequency > 0)

# Merge sentiment scores with word data
word_data_sentiment <- word_data_long |>
  inner_join(sentiments, by = "word")
# Calculate daily sentiment scores
daily_sentiment <- word_data_sentiment %>%
  group_by(day) %>%
  summarize(sentiment_score = sum(frequency * score))

# Merge sentiment and volatility data
volatility_data <- data.frame(day = as.numeric(rownames(daily_sentiment)), volatility = V)
analysis_data <- merge(daily_sentiment, volatility_data, by = "day")

summary(analysis_data)
correlation_result <- cor(analysis_data$sentiment_score, analysis_data$volatility)

print(paste0("Correlation between sentiment and volatility:", correlation_result))
# Plot the results
ggplot(analysis_data, aes(x = sentiment_score, y = volatility)) +
  geom_point() +
  geom_smooth(formula = y ~ x, method = "lm", color = "blue") +
  labs(title = "Correlation between Daily Sentiment and Market Volatility",
       x = "Sentiment Score", y = "Volatility")
```